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 **Rohan Lalchandani** 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 **2024-2025.**

Lab Assistant

Subject Teacher

Mrs. Kajal Joseph

Principal

Head of Department

Dr. Mrs. Shalu Chopra

Project Title: Fitness Workout App **Roll No. 25**

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

Year/Sem/Class : D15A/D15B **A.Y.:** 24-25

Faculty Incharge : Mrs. Kajal Joseph.

Lab Teachers : Mrs. Kajal Joseph.

Email : kajal.jewani@ves.ac.in

Programme Outcomes: The graduate will be able to:

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

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

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

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

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

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

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

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

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

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

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

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

Program specific Outcomes

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

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

Project Title: Fitness Workout App **Roll No. 25**

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			
2.	To design Flutter UI by including common widgets.	LO2			
3.	To include icons, images, fonts in Flutter app	LO2			
4.	To create an interactive Form using form widget	LO2			
5.	To apply navigation, routing and gestures in Flutter App	LO2			
6.	To Connect Flutter UI with fireBase database	LO3			
7.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.	LO4			
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			
9.	To implement Service worker events like fetch, sync and push for E-commerce PWA	LO5			
10.	To study and implement deployment of Ecommerce PWA to GitHub Pages.	LO5			
11.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.	LO6			
12.	Assignment-1	LO1,LO2 ,LO3			
13.	Assignment-2	LO4,LO5 ,LO6			

MAD & PWA Lab
Journal

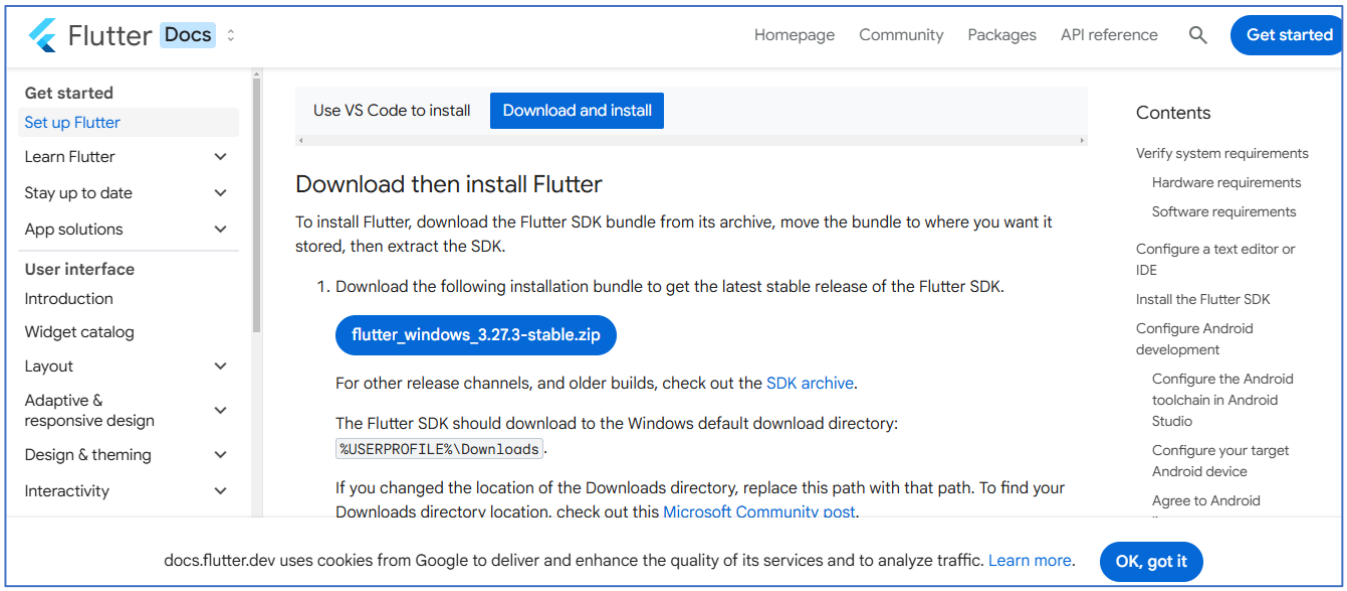
Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	

MAD / PWA Lab

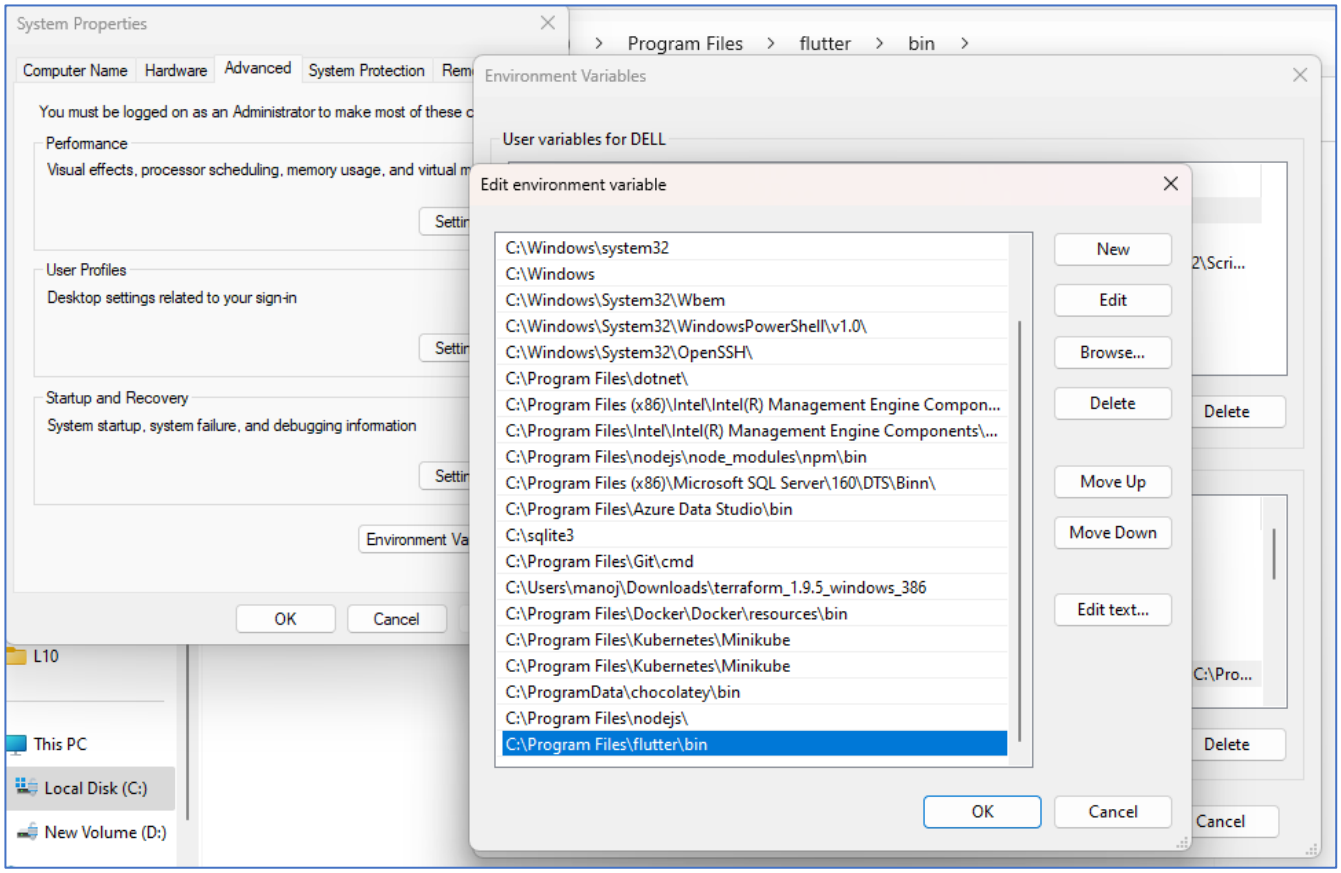
Name: Rohan Lalchandani Class: D15A / B Roll no. 25

Experiment 1 : Installation of flutter.

1. Download flutter SDK and extract the zip file to a directory: https://docs.flutter.dev/get-started/install



2. Copy the bin path in Flutter folder and add it to environment variables.

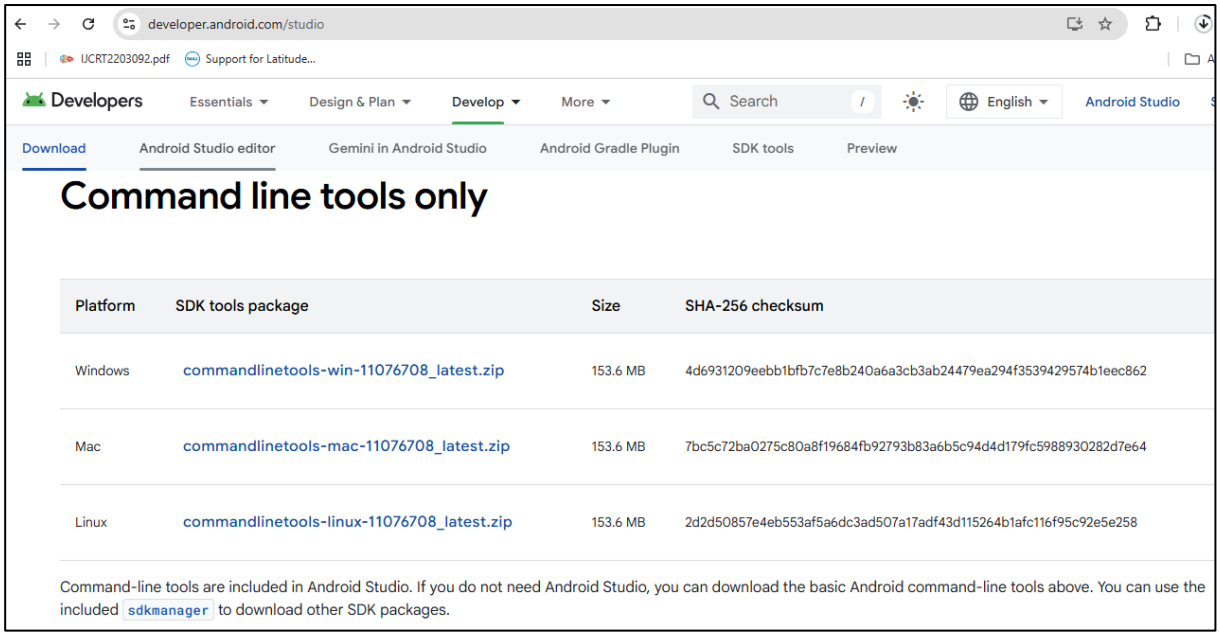


3. Verify installation by running “flutter doctor”.

```
Microsoft Windows [Version 10.0.22631.4751]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\System32>flutter doctor

  A new version of Flutter is available!

  To update to the latest version, run "flutter upgrade".

                  Welcome to Flutter! - https://flutter.dev

  The Flutter tool uses Google Analytics to anonymously report feature usage
  statistics and basic crash reports. This data is used to help improve
  Flutter tools over time.

  Flutter tool analytics are not sent on the very first run. To disable
  reporting, type 'flutter config --no-analytics'. To display the current
  setting, type 'flutter config'. If you opt out of analytics, an opt-out
  event will be sent, and then no further information will be sent by the
  Flutter tool.

  By downloading the Flutter SDK, you agree to the Google Terms of Service.
  The Google Privacy Policy describes how data is handled in this service.

  Moreover, Flutter includes the Dart SDK, which may send usage metrics and
  crash reports to Google.

  Read about data we send with crash reports:
  https://flutter.dev/to/crash-reporting

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

  To disable animations in this tool, use
  'flutter config --no-cli-animations'.
```

```
C:\Windows\System32>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.22631.4751], locale en-IN)
[√] Windows Version (Installed version of Windows is version 10 or higher)
[X] Android toolchain - develop for Android devices
    X Unable to locate Android SDK.
      Install Android Studio from: https://developer.android.com/studio/index.html
      On first launch it will assist you in installing the Android SDK components.
      (or visit https://flutter.dev/to/windows-android-setup for detailed instructions).
      If the Android SDK has been installed to a custom location, please use
      `flutter config --android-sdk` to update to that location.

[√] Chrome - develop for the web
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[!] Android Studio (not installed)
[√] VS Code (version 1.97.1)
[√] Connected device (3 available)
[√] Network resources

! Doctor found issues in 3 categories.
```

4. Install Android SDK Command line tools from https://developer.android.com/studio
   Download the zip file and extract it to specified folder
   Also add the bin path of android sdk to environment variables.

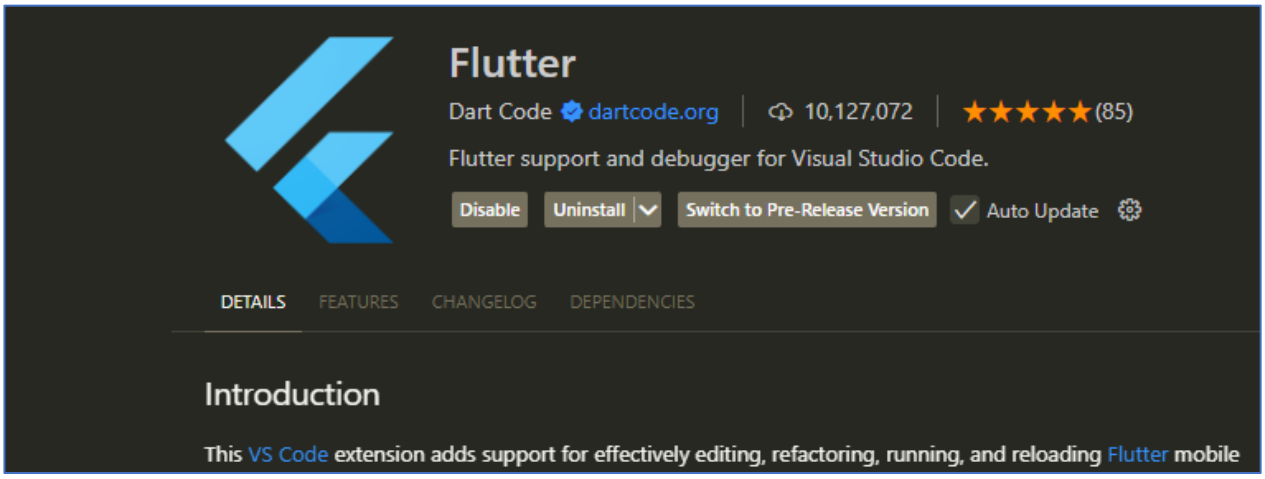


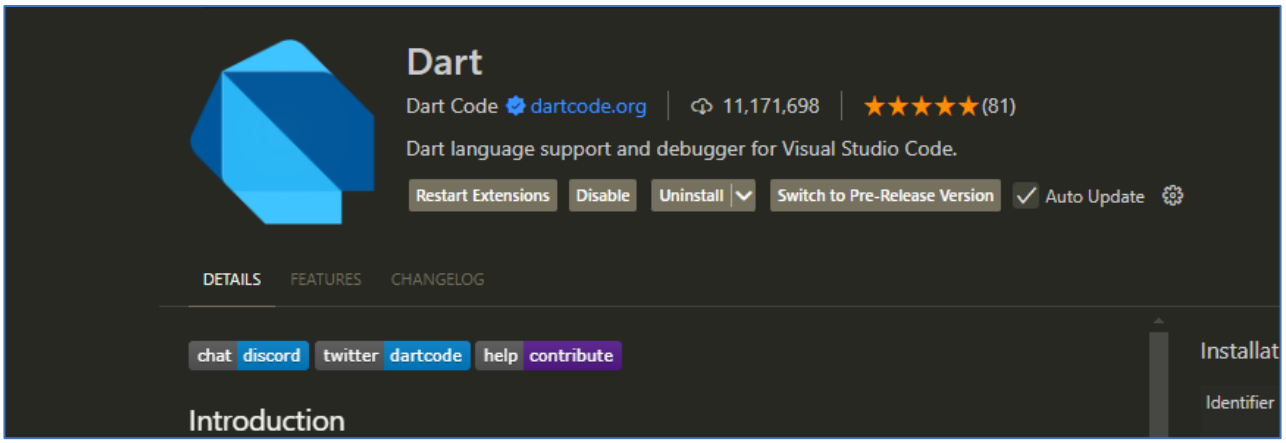
5. Run CMD as administrator and install essential SDK components:
   cd "C:\Program Files\Android SDK\cmdline-tools\latest\bin"
   sdkmanager --install "platform-tools" "platforms;android-34" "build-tools;34.0.0" "emulator"

```
C:\Windows\System32>sdkmanager --install "platform-tools" "platforms;android-34" "build-tools;34.0.0" "emulator"
Warning: Errors during XML parse:
Warning: Additionally, the fallback loader failed to parse the XML.
Warning: Errors during XML parse:
Warning: Additionally, the fallback loader failed to parse the XML.
License android-sdk-license:            ] 10% Computing updates...
---------------------------------------
Terms and Conditions
```

```
14.7 The License Agreement, and your relationship with Google under the License Ag
s conflict of laws provisions. You and Google agree to submit to the exclusive jur
e any legal matter arising from the License Agreement. Notwithstanding this, you a
lent type of urgent legal relief) in any jurisdiction.


January 16, 2019
---------------------------------------
Accept? (y/N): Y
[=======================================] 100% Unzipping... platform-tools/sqlit

C:\Windows\System32>
```

6. Install Flutter and Dart plugin in vs code studio.



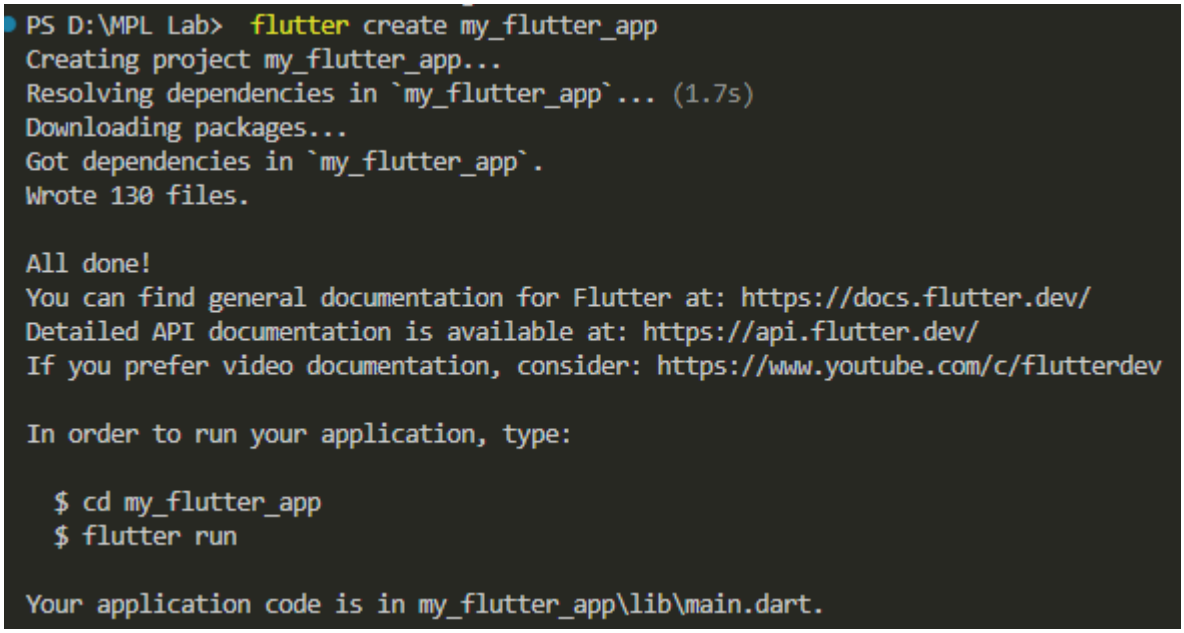


Check all the issues of flutter doctor have been resolved

```
C:\Windows\System32>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.22631.4751], locale en-IN)
[√] Windows Version (Installed version of Windows is version 10 or higher)
[!] Android toolchain - develop for Android devices (Android SDK version 34.0.0)
    ! Some Android licenses not accepted. To resolve this, run: flutter doctor --android-licenses
[√] Chrome - develop for the web
[√] Visual Studio - develop Windows apps (Visual Studio Community 2022 17.13.0)
[!] Android Studio (not installed)
[√] VS Code (version 1.97.1)
[√] Connected device (3 available)
[√] Network resources

! Doctor found issues in 2 categories.
```

Now run this command to create a flutter app and then run

“cd my_flutter_app”



In the directory structure go to lib folder and write the below code in “main.dart” file.

Code:

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


void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(title: Text('Flutter Demo')),
        body: Center(
          child: Text(
```

```
            'Hello Rohan',
            style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),
          ),
        ),
      ),
    );
  }
}
```

Output:

MAD & PWA Lab
Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL EXPERIMENT 2

NAME- Rohan Lalchandani
CLASS- D15A ROLL NO- 25

AIM: To design Flutter UI by including common widgets

1,Introduction

Flutter is an open-source UI framework by Google that enables developers to build cross-platform applications with a single codebase. The UI in Flutter is built using **widgets**, which are the fundamental building blocks of the application.

Widgets in Flutter can be classified into various categories, such as structural widgets, layout widgets, interactive widgets, and styling widgets. These widgets help in designing complex UI components efficiently and responsively.

2.**Types of widgets**

Flutter widgets are classified into the following types based on their functionality:

1. Structural Widgets (Scaffolding Widgets)

These widgets define the basic structure of an application.

- **Scaffold:** Provides a framework with an app bar, body, floating action button, drawer, etc.
- **AppBar:** Displays the top navigation bar with titles, actions, and menus.
- **Drawer:** A side navigation panel for navigating through different screens.

2. Layout Widgets

These widgets help in arranging other widgets on the screen.

- **Container:** A flexible widget for holding and styling child widgets.

- **Column:** Arranges widgets vertically.
- **Row:** Arranges widgets horizontally.
- **Stack:** Allows widgets to be placed on top of each other.
- **Expanded:** Expands widgets to fill available space.

- **SizedBox:** Provides spacing between widgets.

3. Interactive Widgets

These widgets accept user input and interactions.

- **TextField:** Allows users to enter text input.
- **ElevatedButton:** A clickable button with elevation.
- **OutlinedButton:** A button with an outlined border.
- **IconButton:** A clickable icon button.
- **GestureDetector:** Detects gestures like tap, swipe, and drag.
- **Switch:** A toggle button for enabling/disabling options.

4. Styling & Display Widgets

These widgets help in styling and displaying content.

- **Text:** Displays text content.
- **Image:** Displays an image from assets or a URL.
- **Icon:** Displays an icon from Material or custom icons.
- **Card:** A box with elevation and rounded corners for content display.
- **Chip:** A small, compact element that represents information.

5. Scrolling & List Widgets

These widgets handle large lists and scrolling content.

- **ListView:** Displays a scrollable list of items.
- **GridView:** Displays items in a grid format.
- **SingleChildScrollView:** Makes a child widget scrollable.
- **PageView:** Creates swipeable pages.

3. **Commonly used widgets:**

1. Scaffold (Basic Page Structure)

A *Scaffold* provides a basic structure for a screen, including an **AppBar, Floating Action Button, Drawer, and Body**.

2.Container (A Flexible Box for UI Design)

A *Container* is used for styling, padding, and positioning elements.

3.Row and Column (Arranging Widgets)

A *Row* arranges widgets horizontally, while a *Column* arranges them vertically.

4.Stack (Overlapping Widgets)

A Stack places widgets on top of each other.

5.ListView (Scrollable List)

A ListView is used for displaying a scrollable list of items.

6.ElevatedButton (Clickable Button)

Buttons provide user interactions in Flutter.

7.TextField (User Input Field)

A TextField allows users to enter data.

Code:

1. **day_exercises_row.dart:**

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

import 'package:healtho_gym/common/color_extension.dart';

class DayExerciseRow extends StatelessWidget {
  final Map obj;
  final VoidCallback onPressed;

  const DayExerciseRow({super.key, required this.obj, required this.onPressed});

  @override
  Widget build(BuildContext context) {
    return InkWell(
      onTap: onPressed,
      child: Container(
        decoration: BoxDecoration(
          color: TColor.txtBG,
          border: Border.all(color: TColor.board, width: 1),
          borderRadius: BorderRadius.circular(15),
        ),
        child: Column(
          children: [
            Padding(
              padding: const EdgeInsets.symmetric(
                horizontal: 15,
                vertical: 15,
              ),
              child: Row(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  ClipRRect(
                    borderRadius: BorderRadius.circular(
                      10,
                    ),
                    child: ClipRRect(
                      borderRadius: BorderRadius.circular(10),
                      child: Image.asset(
                        obj["image"],
                        width: 60,
                        height: 60,
                        fit: BoxFit.cover,
                      ),
```

```
        ),
      ),
      const SizedBox(
        width: 25,
      ),
      Expanded(
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            Text(
              obj["name"],
              style: TextStyle(
                color: TColor.primaryText,
                fontSize: 14,
              ),
            ),
            const SizedBox(
              height: 4,
            ),
            Row(
              children: [
                SizedBox(
                  width: 80,
                  child: Text(
                    "Sets",
                    style: TextStyle(
                      color: TColor.primaryText,
                      fontSize: 12,
                    ),
                  ),
                ),
                Expanded(
                  child: Text(
                    obj["sets"],
                    style: TextStyle(
                      color: TColor.placeholder,
                      fontSize: 12,
                    ),
                  ),
                )
              ],
            ),
            Row(
              children: [
                SizedBox(
```

```
      width: 80,                    width: 80,
      child: Text(                  child: Text(
        "Reps",                       "Reset",
        style: TextStyle(             style: TextStyle(
          color: TColor.primaryText,    color: TColor.primaryText,
          fontSize: 12,                 fontSize: 12,
        ),                            ),
      ),                            ),
    ),                            ),
    Expanded(                     Expanded(
      child: Text(                  child: Text(
        obj["reps"],                  obj["rest"],
        style: TextStyle(             style: TextStyle(
          color: TColor.placeholder,    color: TColor.placeholder,
          fontSize: 12,                 fontSize: 12,
        ),                            ),
      ),                            ),
    )                             )
  ],                            ],
),                              ),
Row(                            ],
  children: [                   ),
    SizedBox(                   ),
```

```
      const SizedBox(
        width: 30,
      ),
      Image.asset(
        "assets/img/next.png",
        width: 12,
        color: TColor.placeholder,
      )
    ],
  ),
),
Container(
  color: TColor.board,
  height: 2,
),
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 15,
    vertical: 15,
  ),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      Image.asset(
        obj["is_complete"]
            ? "assets/img/check_tick.png"
            : "assets/img/uncheck.png",
        width: 20,
      ),
      const SizedBox(
        width: 8,
      ),
      Text(
        "Mark as completed",
        style: TextStyle(
          color: obj["is_complete"]
              ? const Color(0xff27AE60)
              : TColor.placeholder,
          fontSize: 14,
          fontWeight: FontWeight.w600,
        ),
      )
    ],
  ),
```

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

2. icon_title_subtitle_button.dart

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class IconTitleSubtitleButton extends StatelessWidget {
  final String title;
  final String subtitle;
  final String icon;
  final VoidCallback onPressed;

  const IconTitleSubtitleButton(
      {super.key,
      required this.title,
      required this.subtitle,
      required this.icon,
      required this.onPressed});

  @override
  Widget build(BuildContext context) {
```

```
return Padding(
  padding: const EdgeInsets.symmetric(horizontal: 20),
  child: InkWell(
    onTap: onPressed,
    child: Container(
      padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 15),
      decoration: BoxDecoration(
          color: TColor.btnBG, borderRadius: BorderRadius.circular(15)),
      child: Row(
        children: [
          Image.asset(
            icon,
            width: 50,
            height: 50,
            fit: BoxFit.cover,
          ),
          const SizedBox(
            width: 15,
          ),
          Expanded(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text(
```

```
                    title,
                    style: TextStyle(
                        color: TColor.primaryText,
                        fontSize: 15,
                        fontWeight: FontWeight.w600),
                  ),
                  Text(
                    subtitle,
                    style: TextStyle(
                      color: TColor.primaryText,
                      fontSize: 12,
                    ),
                  ),
                ],
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

3. **number_title_subtitle_button.dart:**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class NumberTitleSubtitleButton extends StatelessWidget {
  final String title;
  final String subtitle;
  final String number;
  final VoidCallback onPressed;

  const NumberTitleSubtitleButton(
      {super.key,
      required this.title,
      required this.subtitle,
      required this.number,
      required this.onPressed});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 10),
      child: InkWell(
        onTap: onPressed,
        child: Container(
```

```
padding: const EdgeInsets.symmetric(
  horizontal: 20,
  vertical: 15,
),
decoration: BoxDecoration(
    color: TColor.txtBG,
    border: Border.all(color: TColor.board, width: 1),
    borderRadius: BorderRadius.circular(15)),
child: Row(
  children: [
    Expanded(
      child: Row(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            number,
            style: TextStyle(
              color: TColor.placeholder,
              fontSize: 17,
              fontWeight: FontWeight.w600,
            ),
          ),
          const SizedBox(
            width: 15,
```

```
    ),
    Expanded(
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            title,
            style: TextStyle(
              color: TColor.primaryText,
              fontSize: 15,
              fontWeight: FontWeight.w600,
            ),
          ),
          Text(
            subtitle,
            style: TextStyle(
              color: TColor.primaryText,
              fontSize: 12,
            ),
          ),
        ],
      ),
    ),
  ],
```

```
              ),
            ),
            Image.asset(
              "assets/img/next.png",
              width: 10,
              color: TColor.secondaryText,
            )
          ],
        ),
      ),
    ),
  );
 }
}
```

4. **radio_button.dart**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class RadioButton extends StatelessWidget {
  final String title;
  final bool isSelect;
  final VoidCallback onPressed;
```

```
const RadioButton(
    {super.key,
    required this.title,
    required this.isSelect,
    required this.onPressed});

@override
Widget build(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.symmetric(
      horizontal: 20,
      vertical: 15,
    ),
    child: InkWell(
      onTap: onPressed,
      child: Row(
        children: [
          Container(
            width: 20,
            height: 20,
            decoration: BoxDecoration(
                color: isSelect ? TColor.primary : TColor.inactive,
                borderRadius: BorderRadius.circular(10)),
          ),
```

```
            const SizedBox(
              width: 20,
            ),
            Expanded(
              child: Text(
                title,
                style: TextStyle(
                  color: TColor.primaryText,
                  fontSize: 13,
                ),
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

5. **round_button.dart:**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

enum RoundButtonType { primary, line }
```

```
class RoundButton extends StatelessWidget {
  final String title;
  final RoundButtonType type;
  final double height;
  final double fontSize;
  final FontWeight fontWeight;
  final double width;
  final String? image;
  final bool isPadding;
  final double radius;
  final VoidCallback onPressed;
  const RoundButton(
      {super.key,
      required this.title,
      this.type = RoundButtonType.primary,
      this.height = 50,
      this.fontSize = 14,
      this.radius = 25,
      this.fontWeight = FontWeight.w600,
      this.width = double.maxFinite,
      this.isPadding = true,
      required this.onPressed, this.image});
```

```
@override
Widget build(BuildContext context) {
  return InkWell(
     borderRadius: BorderRadius.circular(25),
    onTap: onPressed,
    child: Container(
      margin: EdgeInsets.symmetric(horizontal: isPadding ? 20 : 0),
      padding: const EdgeInsets.symmetric(horizontal: 20),
      width: width,
      decoration: BoxDecoration(
          color:
              type == RoundButtonType.primary ? TColor.primary : TColor.txtBG,
          border: type == RoundButtonType.line
              ? Border.all(color: TColor.board)
              : null,
          borderRadius: BorderRadius.circular(radius)),
      height: height,
      child: Row(
        children: [
          if(image != null)
          SizedBox(
            width: 60,
            child: Image.asset(image!, width: 20, height: 20,),
          ),
```

```
            Expanded(
              child: Text(
                title,
                textAlign: TextAlign.center,
                style: TextStyle(
                    color: type == RoundButtonType.primary
                        ? TColor.btnPrimaryText
                        : TColor.btnSecondaryText,
                    fontSize: fontSize,
                    fontWeight: fontWeight),
              ),
            ),
             if(image != null)
             Container(
              width: 60,
             )
          ],
        ),
      ),
    );
  }
}

class RoundSelectButton extends StatelessWidget {
```

```
final String title;
final RoundButtonType type;
final double height;
final double fontSize;
final FontWeight fontWeight;
final double width;
final String? image;
final bool isPadding;
final VoidCallback onPressed;
const RoundSelectButton(
    {super.key,
    required this.title,
    this.type = RoundButtonType.primary,
    this.height = 50,
    this.fontSize = 14,
    this.fontWeight = FontWeight.w600,
    this.width = double.maxFinite,
    this.isPadding = true,
    required this.onPressed,
    this.image});

@override
Widget build(BuildContext context) {
  return InkWell(
```

```
 borderRadius: BorderRadius.circular(25),
onTap: onPressed,
child: Container(
  margin: EdgeInsets.symmetric(horizontal: isPadding ? 20 : 0),
  padding: const EdgeInsets.symmetric(horizontal: 20),
  width: width,
  decoration: BoxDecoration(
      color:
          type == RoundButtonType.primary ? TColor.primary : TColor.txtBG,
      border: type == RoundButtonType.line
          ? Border.all(color: TColor.board)
          : null,
      borderRadius: BorderRadius.circular(25)),
  height: height,
  child: Row(
    children: [
      if (image != null)
        SizedBox(
          width: 60,
          child: Image.asset(
            image!,
            width: 20,
            height: 20,
          ),
```

```
            ),
          Expanded(
            child: Text(
              title,
              textAlign: TextAlign.left,
              style: TextStyle(
                  color: type == RoundButtonType.primary
                      ? TColor.btnPrimaryText
                      : TColor.btnSecondaryText,
                  fontSize: fontSize,
                  fontWeight: fontWeight),
            ),
          ),
          if (image != null)
            Container(
              width: 60,
            )
        ],
      ),
    ),
  );
 }
}
```

6. **round_dropdown.dart**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class RoundDropDown extends StatelessWidget {
  final String hintText;
  final List list;
  final dynamic value;
  final bool isPadding;
  final Function(dynamic)? didChange;

  const RoundDropDown({
    super.key,
    required this.hintText,
    required this.list,
    this.value,
    this.isPadding = false,
    this.didChange,
  });

  @override
  Widget build(BuildContext context) {
    return Container(
```

```
height: 50,
margin: EdgeInsets.symmetric(horizontal: isPadding ? 20 : 0),
padding: const EdgeInsets.symmetric(horizontal: 20),
decoration: BoxDecoration(
    color: TColor.txtBG,
    border: Border.all(color: TColor.board, width: 1),
    borderRadius: BorderRadius.circular(10)),
child: DropdownButtonHideUnderline(
  child: DropdownButton(
    isExpanded: true,
    value: value,
    icon: Image.asset(
      "assets/img/down.png",
      width: 15,
    ),
    hint: Text(
      hintText,
      style: TextStyle(color: TColor.secondaryText, fontSize: 15),
    ),
    items: list
        .map(
          (obj) => DropdownMenuItem(
            value: obj,
```

```
                child: Text(
                  obj["name"] as String? ?? "",
                  style: TextStyle(
                    color: TColor.primaryText,
                  ),
                ),
              ),
            )
            .toList(),
        onChanged: didChange,
      ),
    ),
  );
 }
}
```

7. round_text_field.dart

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class RoundTextField extends StatelessWidget {
  final String hintText;
  final TextEditingController? controller;
  final TextInputType? keyboardType;
```

```
final double radius;
final bool obscureText;
final Widget? right;
final bool isPadding;
final String? icon; // Added icon parameter

const RoundTextField({
  super.key,
  required this.hintText,
  this.controller,
  this.keyboardType,
  this.radius = 25,
  this.obscureText = false,
  this.right,
  this.isPadding = false,
  this.icon, // Added icon parameter
});

@override
Widget build(BuildContext context) {
  return Container(
    height: 50,
    margin: EdgeInsets.symmetric(horizontal: isPadding ? 20 : 0),
```

```
decoration: BoxDecoration(
    color: TColor.txtBG,
    border: Border.all(color: TColor.board, width: 1),
    borderRadius: BorderRadius.circular(radius)),
child: TextField(
  controller: controller,
  keyboardType: keyboardType,
  obscureText: obscureText,
  style: TextStyle(color: TColor.primaryText, fontSize: 16),
  decoration: InputDecoration(
    contentPadding: const EdgeInsets.symmetric(horizontal: 20),
    enabledBorder: InputBorder.none,
    focusedBorder: InputBorder.none,
    hintText: hintText,
    prefixIcon: icon != null // Use the icon as a prefixIcon
        ? Padding(
            padding: const EdgeInsets.all(12.0),
            child: Image.asset(icon!),
          )
        : null,
    suffixIcon: right,
    hintStyle: TextStyle(
      color: TColor.placeholder,
```

```
            fontSize: 15,
            fontWeight: FontWeight.w600,
          ),
        ),
      ),
    );
  }
}
```

8. round_title_value_button.dart

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class RoundTitleValueButton extends StatelessWidget {
  final String title;
  final String value;
  final VoidCallback onPressed;
  const RoundTitleValueButton({
    super.key,
    required this.title,
    required this.value,
    required this.onPressed,
  });
```

```
@override
Widget build(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 15),
    child: InkWell(
      borderRadius: BorderRadius.circular(25),
      onTap: onPressed,
      child: Container(
        padding: const EdgeInsets.symmetric(horizontal: 20),
        decoration: BoxDecoration(
            color: TColor.txtBG,
            border: Border.all(color: TColor.board, width: 1),
            borderRadius: BorderRadius.circular(25)),
        height: 50,
        child: Row(
          children: [
            Expanded(
              child: Text(
                title,
                textAlign: TextAlign.center,
                style: TextStyle(
                  color: TColor.btnSecondaryText,
```

```
                fontSize: 15,
                fontWeight: FontWeight.w600,
              ),
            ),
          ),
          Expanded(
            flex: 2,
            child: Text(
              value,
              textAlign: TextAlign.center,
              style: TextStyle(
                color: TColor.btnSecondaryText,
                fontSize: 15,
                fontWeight: FontWeight.w600,
              ),
            ),
          ),
        ],
      ),
    ),
  );
}
```

}

9. **secrtion_button.dart**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class SectionButton extends StatelessWidget {
  final String title;
  final VoidCallback onPressed;

  const SectionButton(
      {super.key, required this.title, required this.onPressed});

  @override
  Widget build(BuildContext context) {
    return InkWell(
      onTap: onPressed,
      child: Padding(
        padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 8),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.spaceBetween,
          children: [
            Text(
              title,
```

```
            style: TextStyle(
              color: TColor.primaryText,
              fontSize: 15,
              fontWeight: FontWeight.w600,
            ),
          ),
          Text(
            "More",
            style: TextStyle(
              color: TColor.primary,
              fontSize: 15,
            ),
          )
        ],
      ),
    ),
  );
 }
}
```

10. **title_subtitle_button.dart**

```dart
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class TitleSubtitleButton extends StatelessWidget {
  final String title;
  final String subtitle;
  final VoidCallback onPressed;

  const TitleSubtitleButton(
      {super.key,
      required this.title,
      required this.subtitle,
      required this.onPressed});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 10),
      child: InkWell(
        onTap: onPressed,
        child: Container(
          padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 15),
          decoration: BoxDecoration(
```

```
  color: TColor.txtBG,
  border: Border.all(color: TColor.board, width: 1),
  borderRadius: BorderRadius.circular(10),
),
child: Column(
  crossAxisAlignment: CrossAxisAlignment.center,
  children: [
    Text(
      title,
      style: TextStyle(
          color: TColor.primaryText,
          fontSize: 14,
          fontWeight: FontWeight.w600),
    ),
    Text(
      subtitle,
      style: TextStyle(
        color: TColor.primaryText,
        fontSize: 12,
      ),
    ),
  ],
),
```

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

11. title_value_row.dart

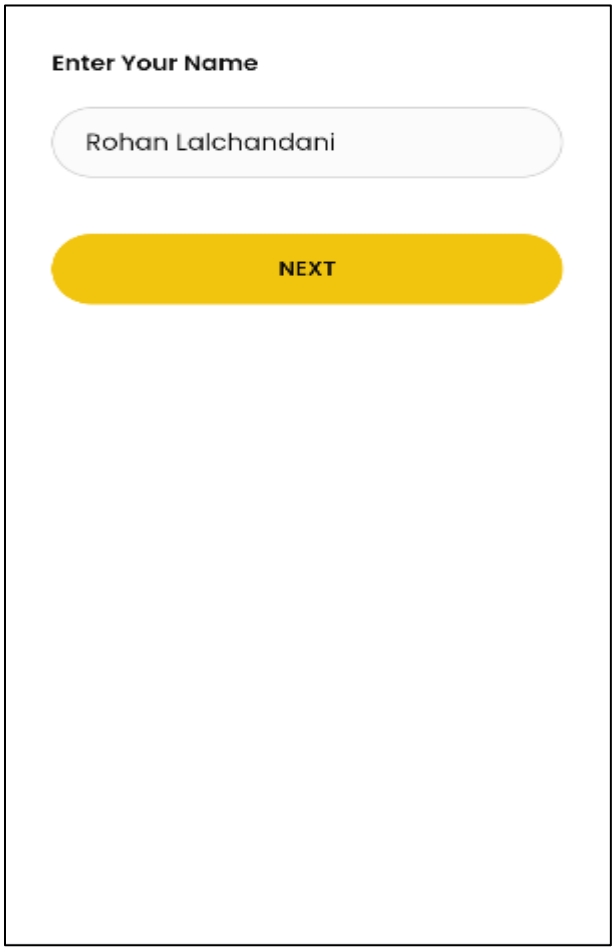
```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class TitleValueRow extends StatelessWidget {
  final String title;
  final String value;
  const TitleValueRow({super.key, required this.title, required this.value});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.symmetric(vertical: 15),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
```

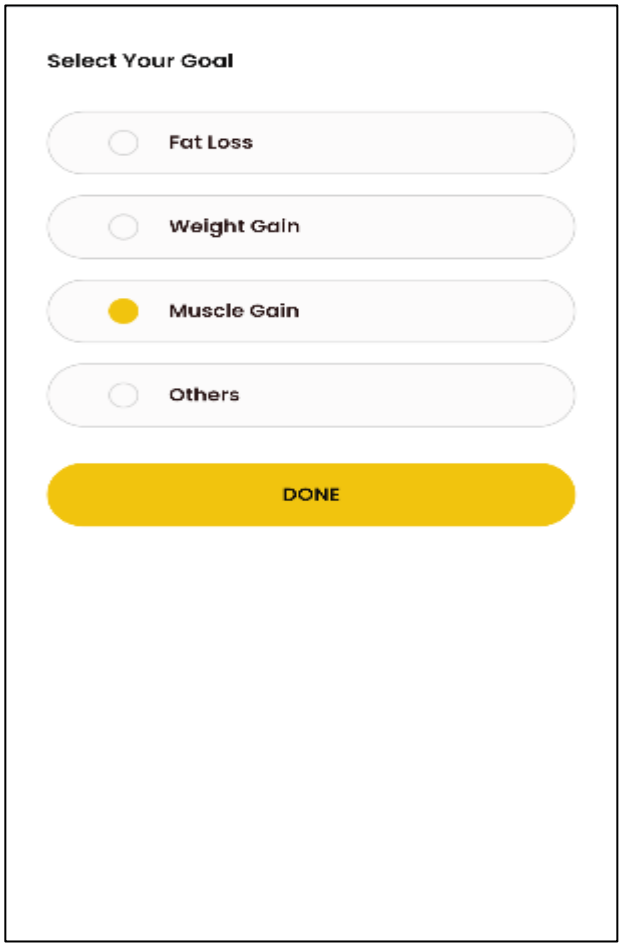
```
            title,
            style: TextStyle(
              color: TColor.placeholder,
              fontSize: 14,
            ),
          ),
          const SizedBox(
            height: 4,
          ),
          Text(
            value,
            style: TextStyle(
              color: TColor.primaryText,
              fontSize: 14,
              fontWeight: FontWeight.w600,
            ),
          ),
        ],
      ),
    );
  }
}
```

12. **top_tab_button.dart**

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class TopTabButton extends StatelessWidget {
  final String title;
  final bool isSelect;
  final VoidCallback onPressed;

  const TopTabButton(
      {super.key,
      required this.title,
      required this.isSelect,
      required this.onPressed});

  @override
  Widget build(BuildContext context) {
    return InkWell(
      onTap: onPressed,
      child: Container(
        height: 40,
        padding: const EdgeInsets.symmetric(horizontal: 20),
        decoration: BoxDecoration(
```

```
            border: Border(
                bottom: BorderSide(
                    color: isSelect ? TColor.primary : Colors.transparent,
                    width: 3))),
        alignment: Alignment.center,
        child: Text(
          title,
          style: TextStyle(
              color: isSelect ? TColor.primary : Colors.white,
              fontSize: 14,
              fontWeight: FontWeight.normal),
        ),
      ),
    );
  }
}
```

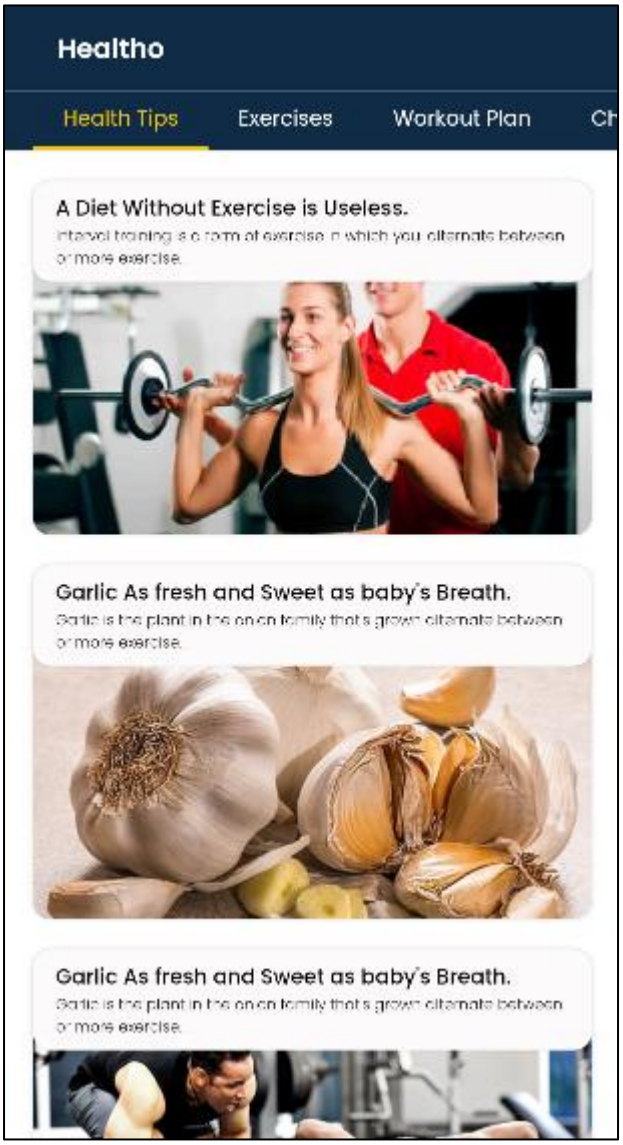

Enter Your Name
Rohan Lalchandani
NEXT

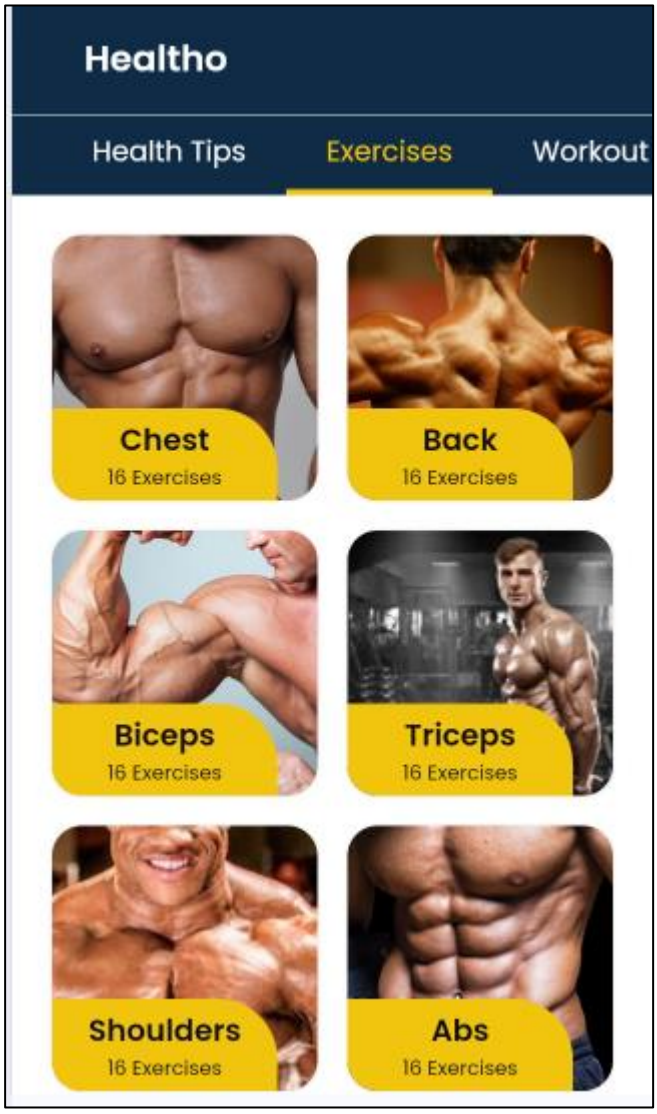

Select Your Goal
Fat Loss
Weight Gain
Muscle Gain
Others
DONE

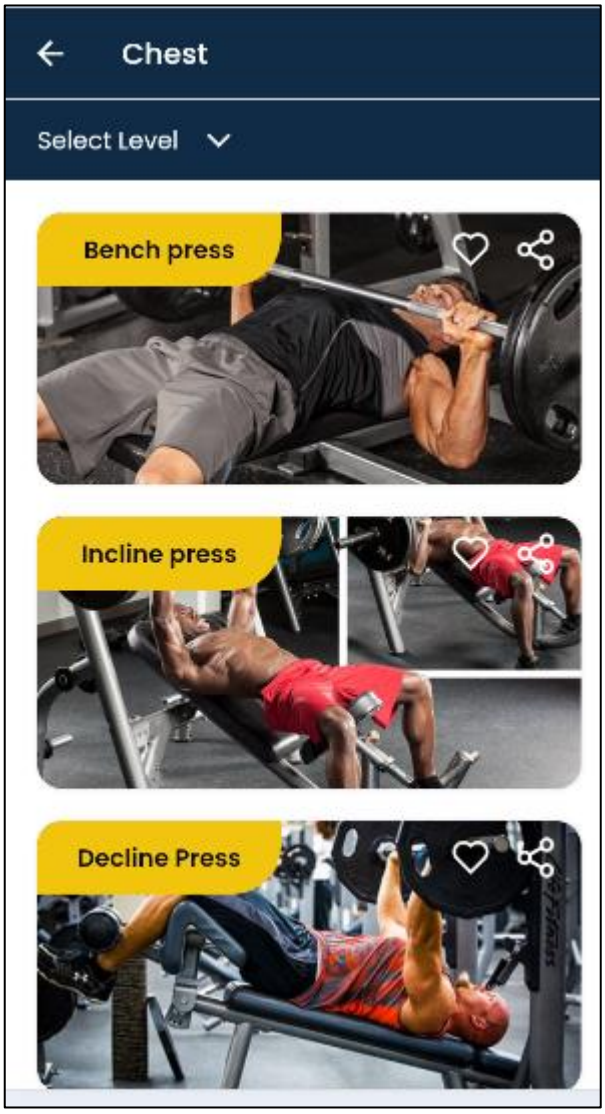
Enter Your Physique

Age	21 Yrs
Height	6 Ft 4 Inch
Weight	40 KG
Level	Beginner

Confirm Detail




Healtho
Health Tips
Exercises
Workout
Chest
16 Exercises
Back
16 Exercises
Biceps
16 Exercises
Triceps
16 Exercises
Shoulders
16 Exercises
Abs
16 Exercises


Chest
Select Level
Bench press
Incline press
Decline Press

Healtho

Health Tips | Exercises | Workout Plan

Find a Workout Plan
Perfect Workout plan that fulfill your fitness goal

My Plan

Create New Plan
Customise workout plans as per your Need

Muscle Building More

Gain Strength More

← Muscle Building

Introduction >
This is a beginner quick start guide that will move you from day 1 to day 60, providing you with starting training and instructions...

30% Complete

01 **Week**
This is a beginner quick start..... >

02 **Week**
This is a beginner quick start..... >

03 **Week**
This is a beginner quick start..... >

MAD & PWA Lab
Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 3

Name: Rohan Lalchandani

Class: D15A Roll no: 25

AIM- To include images, fonts in flutter app.

Theory:

Images are an essential part of UI design, and Flutter supports adding both local and network images.

A) Local images can be stored in the project directory and loaded into the app.

Steps to add Local images:

- Create an assets folder in the root directory.
- Store images inside the assets folder.
- Declare assets in pubspec.yaml under the flutter section: flutter:
  assets:
  - assets/image1.png
  - assets/images/image2.jpg

B) Network Images

Flutter allows displaying images from the internet using Image.network():

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

Font Awesome provides a vast collection of scalable vector icons that behave like fonts. These icons can be used in Flutter via the font_awesome_flutter package, which integrates Font Awesome's font-based icons seamlessly into the app.

SYNTAX

1)Create an assets folder for Local images.

Declare assets in pubspec.yaml file.

flutter: assets:

- assets/image1.png

- assets/images/image2.jpg

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

2)If using network Images

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

3)Install fontawesome package in flutter

Add this dependency in pubspec.yaml file

dependencies: font_awesome_flutter:

^10.7.0

Run flutter pub get

FaIcon(FontAwesomeIcons.heart, size: 50, color: Colors.red)

Widget properties

1)image

- width: Sets image width.
- height: Sets image height.
- fit: Controls how image fits (e.g., BoxFit.cover, BoxFit.fill).
- alignment: Aligns the image inside the container.
- color: Applies a color filter.
- opacity: Controls image transparency.
- loadingBuilder: Handles loading states.
- errorBuilder: Handles image load errors.

Example

```
Image.network(
  'https://example.com/image.jpg',  width: 100,  height: 100,
fit: BoxFit.contain,  loadingBuilder: (context, child, progress) {
return progress == null ? child : CircularProgressIndicator();
  },
  errorBuilder: (context, error, stackTrace) {
return Icon(Icons.error);
  },
)
```

2)font

- size: Adjusts icon size.
- color: Sets icon color.
- semanticLabel: Adds an accessibility label for screen readers.

Example:
FaIcon(
  FontAwesomeIcons.heart,
  size: 50,   // Sets icon size   color: Colors.red, // Sets
icon color   semanticLabel: 'Heart Icon', // Provides
accessibility label
)

Code:

pubspec.yaml Fonts Dependencies:

```
fonts:

    - family: Poppins
      fonts:
        - asset: assets/font/Poppins-Regular.ttf
        - asset: assets/font/Poppins-Medium.ttf
          weight: 500
        - asset: assets/font/Poppins-Light.ttf
          weight: 300
        - asset: assets/font/Poppins-SemiBold.ttf
          weight: 600
        - asset: assets/font/Poppins-Bold.ttf
          weight: 700
```

namescreen.dart

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';
import 'package:healtho_gym/common_widget/round_button.dart';
import 'package:healtho_gym/common_widget/round_text_field.dart';
import 'package:healtho_gym/screen/login/goal_screen.dart';

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

  @override
  State<NameScreen> createState() => _NameScreenState();
}

class _NameScreenState extends State<NameScreen> {
```

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 30),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              const SizedBox(
                height: 30,
              ),
              Text(
                "Enter Your Name",
                style: TextStyle(
                  color: TColor.primaryText,
                  fontSize: 15,
                  fontWeight: FontWeight.w600,
                ),
              ),
              const SizedBox(
                height: 20,
              ),
              const RoundTextField(
                hintText: "i.e code for any",
              ),
              const SizedBox(
                height: 40,
              ),
              RoundButton(
                  title: "NEXT",
                  isPadding: false,
                  onPressed: () {
                    context.push(const GoalScreen());
                  }),
              const Spacer()
            ],
          ),
        ),
      ),
    );
  }
}
```

goalscreen.dart:

```
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';
import 'package:healtho_gym/common_widget/round_button.dart';
import 'package:healtho_gym/screen/home/setting/setting_screen.dart';
import
'package:healtho_gym/screen/home/top_tab_view/top_tab_view_screen.dart';
import 'package:healtho_gym/screen/login/physique_screen.dart';

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

  @override
  State<GoalScreen> createState() => _GoalScreenState();
}

class _GoalScreenState extends State<GoalScreen> {

  String selectName = "";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 30),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              const SizedBox(
                height: 30,
              ),
              Text(
                "Select Your Goal",
                style: TextStyle(
                  color: TColor.primaryText,
                  fontSize: 15,
                  fontWeight: FontWeight.w600,
                ),
              ),
              const SizedBox(
```

```
              height: 20,
            ),
            Column(
              children: ["Fat Loss", "Weight Gain", "Muscle Gain", "Others"]
                  .map((name) {
                return Padding(
                  padding: const EdgeInsets.symmetric(vertical: 8),
                  child: RoundSelectButton(title: name, type:
RoundButtonType.line, isPadding: false,
                  image: selectName == name ? "assets/img/radio_select.png" :
"assets/img/radio_unselect.png" ,
                   onPressed: (){

                    setState(() {
                      selectName = name;
                    });

                  }),
                );
              }).toList(),
            ),
            const SizedBox(
              height: 20,
            ),
            RoundButton(title: "DONE", isPadding: false, onPressed: () {
              context.push(const PhysiqueScreen());
            }),
            const Spacer()
          ],
        ),
      ),
    ),
  );
 }
}
```

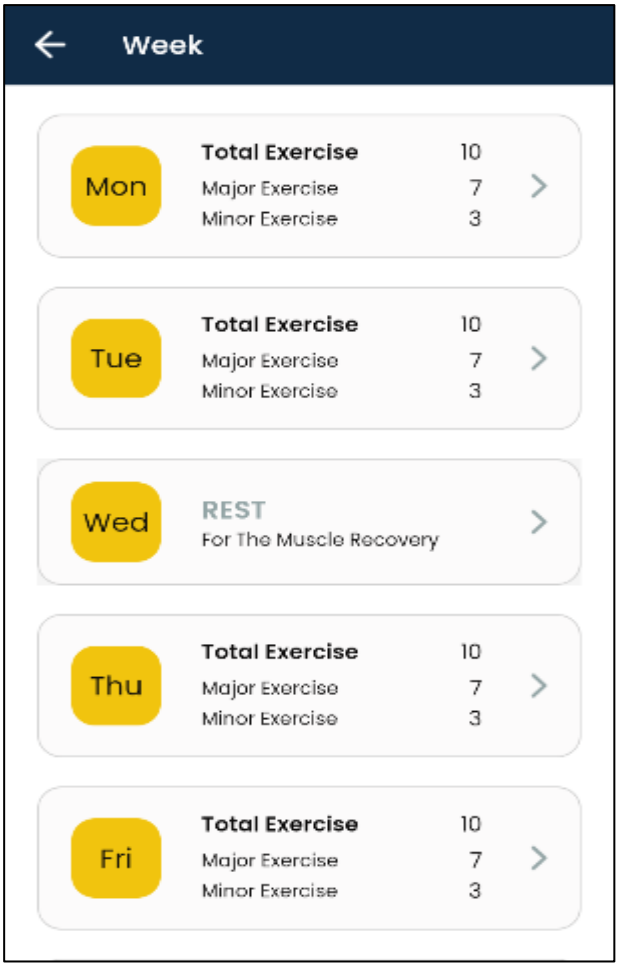
selectage_screen.dart:

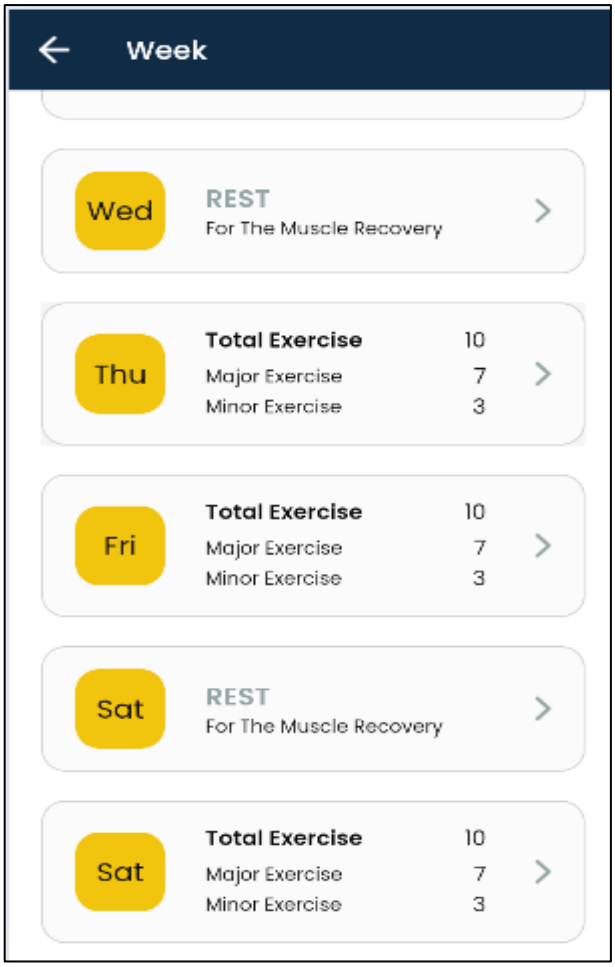
```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:healtho_gym/common/color_extension.dart';

class SelectAgeScreen extends StatefulWidget {
  final Function(dynamic) didChange;
```

```
  const SelectAgeScreen({super.key, required this.didChange});

  @override
  State<SelectAgeScreen> createState() => _SelectAgeScreenState();
}

class _SelectAgeScreenState extends State<SelectAgeScreen> {
  List valueArr = [];

  @override
  void initState() {
    // TODO: implement initState
    super.initState();

    for (var i = 1; i < 120; i++) {
      valueArr.add({"name": "$i", "value": i});
    }
  }

  @override
  Widget build(BuildContext context) {
    return Container(
      width: context.width,
      height: context.height,
      color: Colors.black45,
      alignment: Alignment.center,
      child: Container(
        width: context.width * 0.6,
        decoration: BoxDecoration(
          color: Colors.white,
          borderRadius: BorderRadius.circular(
            20,
          ),
        ),
        padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 20),
        child: Column(
          mainAxisSize: MainAxisSize.min,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              "Select your Age",
              style: TextStyle(
                color: TColor.primaryText,
                fontSize: 15,
```

```
                fontWeight: FontWeight.w600,
              ),
            ),
            const SizedBox(
              height: 25,
            ),
            SizedBox(
              height: 200,
              child: Row(
                children: [
                  Expanded(
                    child: CupertinoPicker(
                      itemExtent: 32,
                      onSelectedItemChanged: (value) {
                        widget.didChange(valueArr[value]["name"]);
                      },
                      children: List<Widget>.generate(valueArr.length, (index) {
                        var obj = valueArr[index];

                        return Text("${obj["name"]}");
                      }),
                    ),
                  )
                ],
              ),
            )
          ],
        ),
      ),
    );
  }
}
```

Output:


Week
Mon
Total Exercise 10
Major Exercise 7
Minor Exercise 3
Tue
Total Exercise 10
Major Exercise 7
Minor Exercise 3
Wed
REST
For The Muscle Recovery
Thu
Total Exercise 10
Major Exercise 7
Minor Exercise 3
Fri
Total Exercise 10
Major Exercise 7
Minor Exercise 3


Week
Wed
REST
For The Muscle Recovery
Thu
Total Exercise 10
Major Exercise 7
Minor Exercise 3
Fri
Total Exercise 10
Major Exercise 7
Minor Exercise 3
Sat
REST
For The Muscle Recovery
Sat
Total Exercise 10
Major Exercise 7
Minor Exercise 3

Healtho

Plan | Challenges | Trainers | Dietician

Ashish Chutake
Fitness and Physiotherapy
★★★★★ Mumbai

Ann Mathewys
Weight Loss
★★★★★ Nagpur

Lalit Kalambe
Fitness and Physiotherapy
★★★★★ Mumbai

Aditya Khobragade
power gaining
★★★★★ Bangalore

Ashish Chutake
Fitness and Physiotherapy
★★★★★ Chennai

Healtho

Plan | Challenges | Trainers | Dietician

Ashish Chutake
Fitness and Physiotherapy
★★★★★ Mumbai

Ann Mathewys
Weight Loss
★★★★★ Nagpur

Lalit Kalambe
Fitness and Physiotherapy
★★★★★ Mumbai

Aditya Khobragade
power gaining
★★★★★ Bangalore

Ashish Chutake
Fitness and Physiotherapy
★★★★★ Chennai

← Bench Press

Bench Press

1) Lie back on a flat bench. Using a medium width grip, lift the bar from the rack and hold it straight over you with your arms locked. This will be your starting position.

2) From the starting position, breathe in and begin coming down slowly until the bar touches your middle chest.

3) After a brief pause, push the bar back to the starting position as you breathe out.

Equipment Required

Barbell, Bench , Plate, Lock

Target Muscle

Chest, Shoulder, Triceps

MAD & PWA Lab

Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 4

Name: Rohan Lalchandani

Class: D15A Roll no: 25

AIM- To create an interactive form using form widget

THEORY-

Forms are essential components of web and mobile applications, allowing users to input and submit data. An interactive form enhances user experience by providing real-time validation, user-friendly input fields, and seamless data handling.

A Form Widget is a structured way to manage user input, validate data, and handle submissions efficiently. It provides an interactive interface for users to enter and modify information.

Key Features of Interactive Forms

- User Input Fields: Text fields, dropdowns, checkboxes, radio buttons, and other input elements.
- Real-time Validation: Ensures correct data format before submission.
- Error Handling: Displays messages for invalid inputs.
- Data Submission: Sends user input to a backend or local storage for further processing.
- Dynamic Updates: Auto-fills or adjusts form fields based on user selections.

Components of Form Widget

- Form Container: Wraps all input fields.
- Input Fields: Text fields, number fields, password inputs, email inputs, etc.
- Buttons: Submit and reset buttons to process or clear input.
- Validation Mechanisms: Ensures valid input before submission.

SYNTAX

```
Form(   key: formKey, // Unique key to manage form state

child: Column(     children: [       TextFormField(

decoration: InputDecoration(labelText: "Enter your name"),
```

```
validator: (value) {          if (value == null || value.isEmpty) {
return "This field cannot be empty";
        }
return null;
      },
    ),
    SizedBox(height: 10),
ElevatedButton(        onPressed: () {
if (formKey.currentState!.validate()) {
          // Perform form submission action
        }
      },
      child: Text("Submit"),
    ),
  ],
 ),
)
```

Widget Properties

1)key

- Used to uniquely identify the Form widget.
- Typically assigned a GlobalKey<FormState> to manage validation and submissions.

Example, final _formKey = GlobalKey<FormState>();

```
Form(   key: _formKey,   child:
Column(     children: [ /* Form fields go
here */ ],
  ),
);
```

2)child

- Defines the content inside the Form, usually containing form fields like TextFormField, DropdownButtonFormField, etc.

Example:

```
Form(    child:
Column(
children: [
      TextFormField(),
      ElevatedButton(onPressed: () {}, child: Text("Submit")),
    ],
  ),
);
```

3)onchanged

- A callback function that gets triggered when any field inside the form changes.
- Can be used to update state based on form input.

Example:

```
Form(   onChanged:
() {     print("Form data
changed!");
  },
  child: TextFormField(),
);
```

CODE

The introduction page

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

```
'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:myapp/pages/createaccount.dart'; import
'package:myapp/pages/loginpage.dart';


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

 @override
 Widget build(BuildContext context) {     return Scaffold(
body: Center(         child: Padding(           padding: const
EdgeInsets.symmetric(horizontal: 30),           child:
Column(             mainAxisAlignment:
MainAxisAlignment.center,
          children: [
            // X logo at the top center               const
FaIcon(FontAwesomeIcons.twitter, size: 50, color: Colors.blue),
const SizedBox(height: 50),

            // Main Text
            Text(
              "See what's happening in the world right now",
textAlign:  TextAlign.center,                              style:
GoogleFonts.roboto(                              fontSize:  24,
fontWeight: FontWeight.bold,
              ),
            ),
            const SizedBox(height: 30),

            // Continue with Google Button
```

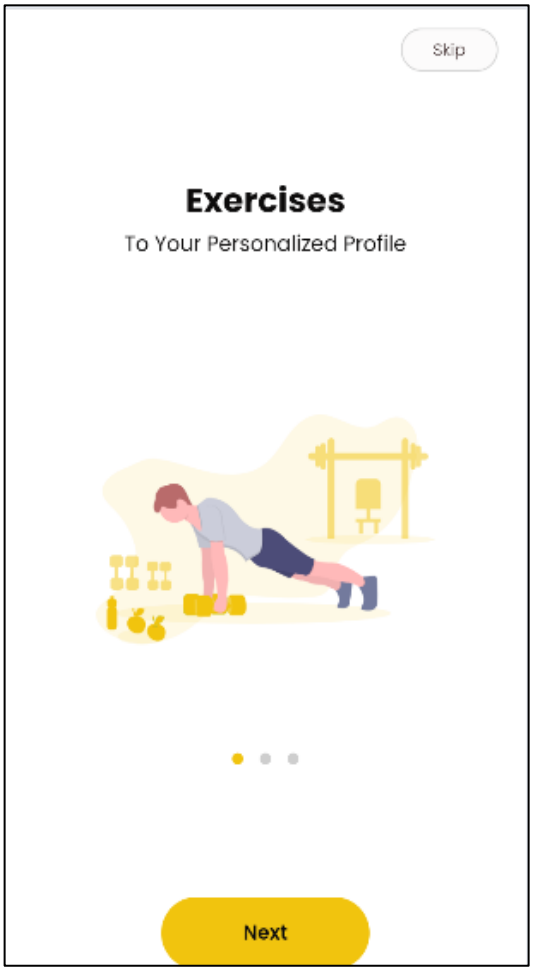
```
            // Create Account Button
ElevatedButton(
onPressed: () {
Navigator.push(
context,
                  MaterialPageRoute(builder: (context) => const CreateAccount()),
                );
              },
              style: ElevatedButton.styleFrom(
                backgroundColor: Colors.blue,                  foregroundColor:
Colors.white,                  padding: const EdgeInsets.symmetric(vertical: 15,
horizontal: 50),                  shape: RoundedRectangleBorder(
borderRadius: BorderRadius.circular(30),
                ),
              ),
              child: const Text("Create account"),
            ),
            const SizedBox(height: 20),

            // Already have an account? Log
in              TextButton(
onPressed: () {
                Navigator.push(context,
                 MaterialPageRoute(builder: (context) => const LoginPage()),
                 );
              },
child: const Text(
                "Have an account already? Log in",
style: TextStyle(color: Colors.blue),
```

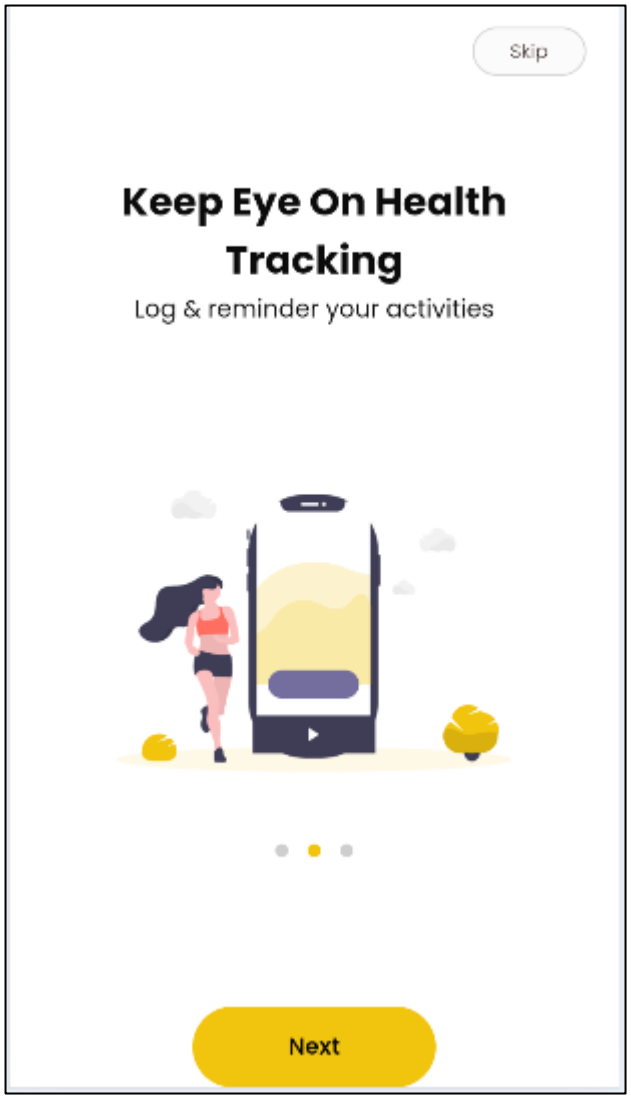
```
            ),
          ),
        ],
      ),
    ),
  ),
 );
 }
}
```

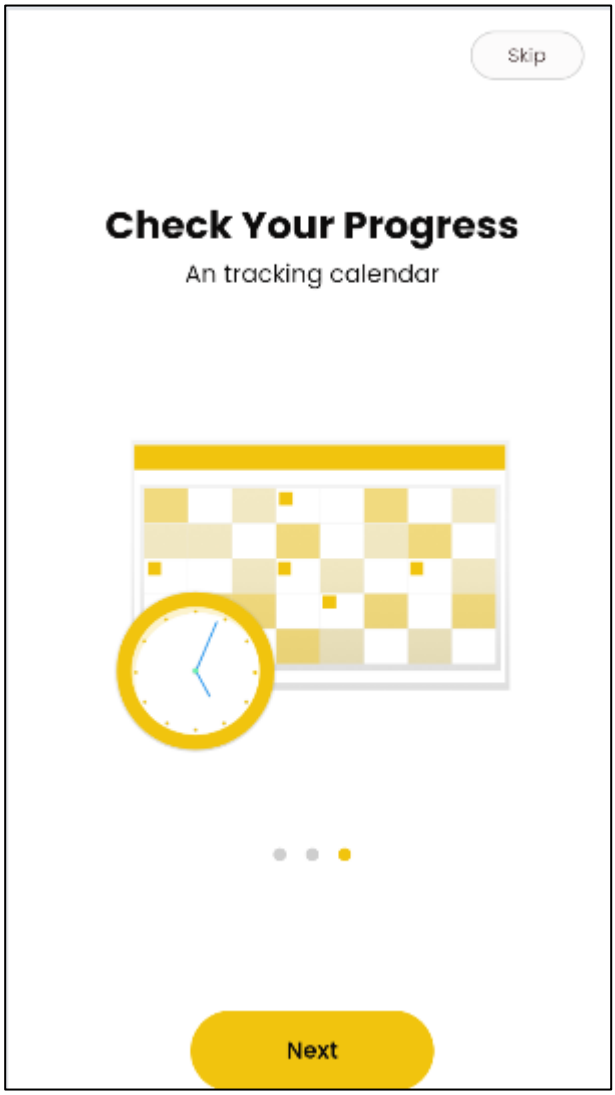
OUTPUT :

Splash Screen

Onboarding Screen







Create Account Page

CODE

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

'package:font_awesome_flutter/font_awesome_flutter.dart';

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

  @override
  _CreateAccountState createState() => _CreateAccountState();
}

class _CreateAccountState extends State<CreateAccount>
{   final _formKey = GlobalKey<FormState>();   final
_usernameController = TextEditingController();   final
_emailController = TextEditingController();   final
_dobController = TextEditingController();   final
_passwordController = TextEditingController();

  @override
  Widget build(BuildContext context) {       return Scaffold(
appBar: AppBar(         leading: IconButton(          icon: const
Icon(Icons.arrow_back, color: Colors.black),         onPressed:
() {
           Navigator.pop(context);
         },
       ),
       centerTitle: true,         title: const FaIcon(FontAwesomeIcons.twitter,
color: Colors.blue, size: 30),         backgroundColor: Colors.transparent,
elevation: 0,
     ),
     body: Padding(         padding: const
EdgeInsets.all(20),         child: Form(          key:
```

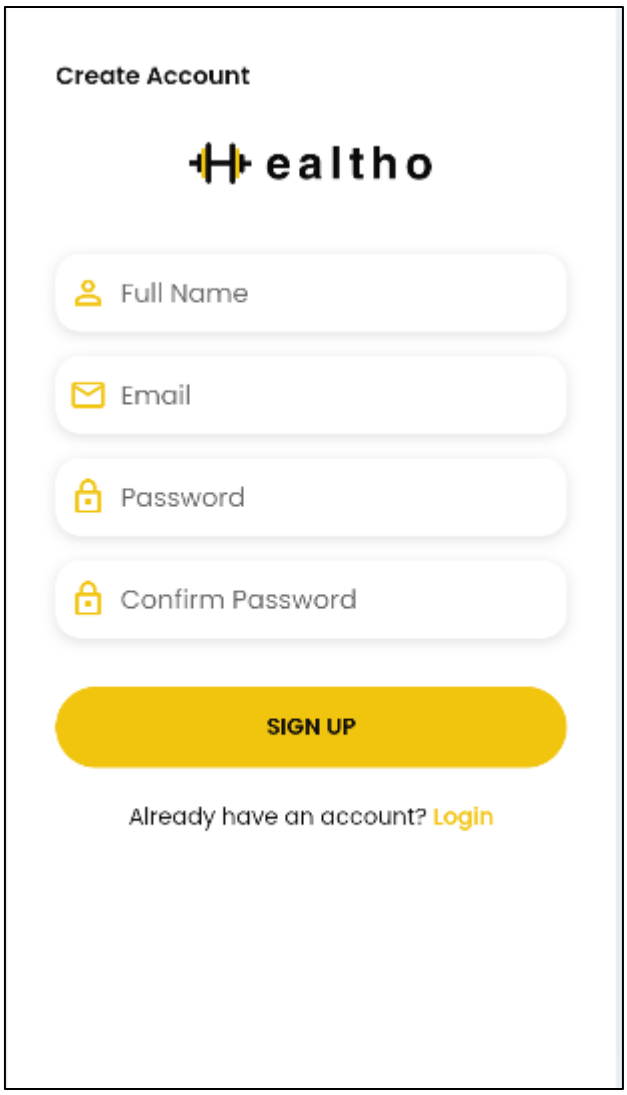
```
_formKey,          child: Column(
crossAxisAlignment: CrossAxisAlignment.start,
          children: [                  // Username                  const Text(
"Username",                  style: TextStyle(fontSize: 16, fontWeight:
FontWeight.bold),
            ),
            TextFormField(                controller: _usernameController,
decoration: InputDecoration(                  hintText: "Enter your username",
border: OutlineInputBorder(borderRadius: BorderRadius.circular(10)),
              ),
              validator: (value) {                  if
(value == null || value.isEmpty) {
return "Username cannot be empty";
                }
                return null;
              },
            ),
            const SizedBox(height: 15),

            // Email            const Text(                "Email",                style:
TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
            ),
            TextFormField(                controller: _emailController,
decoration: InputDecoration(                  hintText: "Enter your email",
border: OutlineInputBorder(borderRadius: BorderRadius.circular(10)),
              ),
              validator: (value) {                  if
(value == null || value.isEmpty) {
return "Email cannot be empty";                  }
```

```
else if (!value.contains("@")) {
return "Enter a valid email with @";
            }
return null;
            },
          ),
          const SizedBox(height: 15),

          // Date of Birth
const Text( "Date of
Birth", style:
TextStyle(fontSize: 16,
fontWeight:
FontWeight.bold),
          ),
          TextFormField( controller: _dobController,
decoration: InputDecoration( hintText: "DD/MM/YYYY",
border: OutlineInputBorder(borderRadius: BorderRadius.circular(10)),
            ),
            keyboardType: TextInputType.datetime, validator: (value) {
final dobPattern = RegExp(r"^(0[1-9]|[12][0-9]|3[01])/(0[1-9]|1[0-2])/\d{4}$");
if (value == null || value.isEmpty) { return "Date of birth cannot be empty";
} else if (!dobPattern.hasMatch(value)) { return "Enter a valid date in
DD/MM/YYYY format";
            }
return null;
            },
          ),
          const SizedBox(height: 15),
```

```
            // Password                  const Text(                  "Password",
style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
            ),
            TextFormField(
controller: _passwordController,
obscureText: true,                decoration:
InputDecoration(                  hintText:
"Enter your password",
border:
OutlineInputBorder(borderRadius:
BorderRadius.circular(10)),
              ),
              validator: (value) {                  if (value == null ||
value.isEmpty) {                    return "Password cannot be
empty";                  } else if (value.length < 6) {
return "Password must be at least 6 characters";
                }
return null;
              },
            ),
          ],
        ),
      ),
    ),

    // Floating Action Button for "Next"       floatingActionButton: Padding(
padding: const EdgeInsets.only(bottom: 30), // Adjust to move the button up
child: FloatingActionButton(          onPressed: () {             if
(_formKey.currentState!.validate()) {               print("Username:
${_usernameController.text}");              print("Email:
```

```
${_emailController.text}");              print("DOB: ${_dobController.text}");
print("Password: ${_passwordController.text}");

          }
        },
        backgroundColor: Colors.blue,          child: const
Icon(Icons.arrow_forward, color: Colors.white),
      ),
    ),
    floatingActionButtonLocation: FloatingActionButtonLocation.endFloat,
  );}}
```

OUTPUT



Login Page

CODE

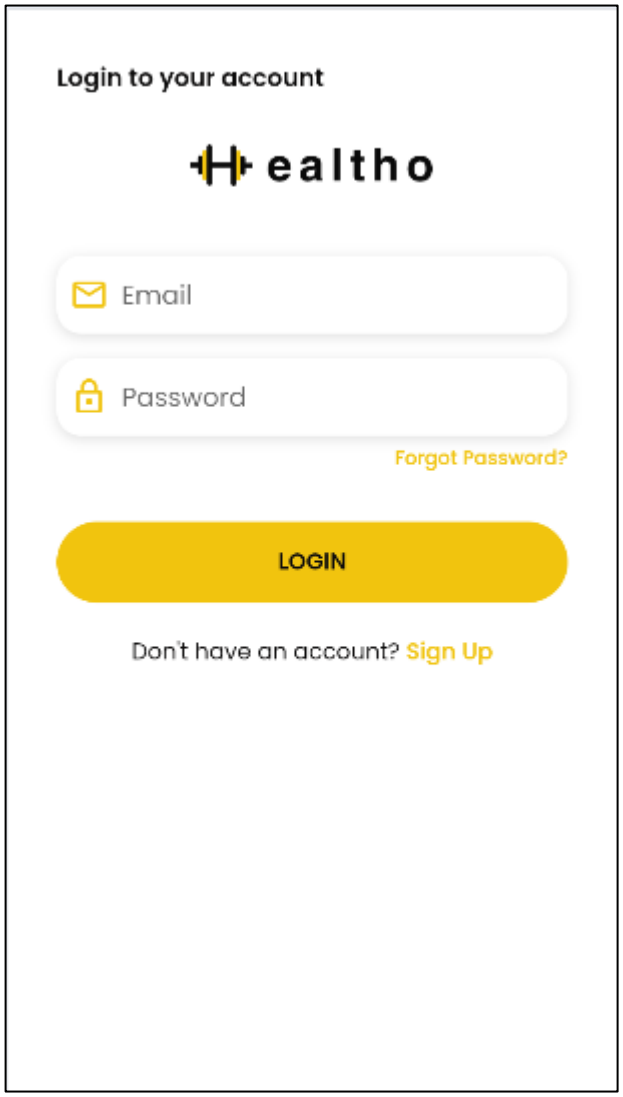
```
import 'package:flutter/material.dart'; import
'package:font_awesome_flutter/font_awesome_flutter.dart';

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

 @override
 _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
final _formKey = GlobalKey<FormState>();   final
_emailController = TextEditingController();   final
_passwordController = TextEditingController();

 @override
 Widget build(BuildContext context) {
return Scaffold(           appBar: AppBar(
leading: IconButton(              icon: const
Icon(Icons.arrow_back),         onPressed:
() {
          Navigator.pop(context);
        },
      ),
    ),
    body: Padding(         padding: const
EdgeInsets.symmetric(horizontal: 30),         child: Form(
```

```
key: _formKey,          child: Column(
crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            // Twitter
Logo              Center(
child: FaIcon(
                FontAwesomeIcons.twitter,
                size: 50,
color: Colors.blue,
              ),
            ),
            const SizedBox(height: 40),

            // Email TextField               const Text(                  "Email",
style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
            ),
            TextFormField(               controller:
_emailController,               decoration:
InputDecoration(                border:
OutlineInputBorder(                   borderRadius:
BorderRadius.circular(10),
              ),
              hintText: "Enter your email",
            ),
            validator: (value) {                if
(value == null || value.isEmpty) {
return "Email cannot be empty";                  }
else if (!value.contains("@")) {
return "Enter a valid email with @";
```

```
            }
return null;
            },
          ),
          const SizedBox(height: 20),

          // Password TextField          const Text(            "Password",
style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),
          ),
          TextFormField(              controller:
_passwordController,              obscureText: true,
decoration: InputDecoration(                border:
OutlineInputBorder(                borderRadius:
BorderRadius.circular(10),
              ),
              hintText: "Enter your password",
            ),
            validator: (value) {                if
(value == null || value.isEmpty) {
return "Password cannot be empty";
} else if (value.length < 6) {                return
"Password must be at least 6 characters";
            }
return null;
            },
          ),
          const SizedBox(height: 40),

          // Next Button aligned at bottom-right          Align(
alignment:  Alignment.bottomRight,                child:
```

```
ElevatedButton(                onPressed: () {                if
(_formKey.currentState!.validate()) {                // Form is
valid, proceed with login logic                print("Email:
${_emailController.text}");                print("Password:
${_passwordController.text}");
                }
              },
              style: ElevatedButton.styleFrom(                backgroundColor:
Colors.blue,                foregroundColor: Colors.white,
padding: const EdgeInsets.symmetric(vertical: 15, horizontal: 30),
shape: RoundedRectangleBorder(                borderRadius:
BorderRadius.circular(30),
                ),
              ),
              child: const Text("Next"),
            ),
          ),
          const SizedBox(height: 30),
        ],
      ),
    ),
  );
 }
}
```

OUTPUT:


Login to your account
ealtho
Email
Password
Forgot Password?
LOGIN
Don't have an account? Sign Up

MAD & PWA Lab

Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	

MPL Experiment 4

Name: Rohan Lalchandani

Class: D15A Roll no: 25

AIM-To apply navigation, routing and gestures in Flutter App

THEORY-

Navigation in Flutter allows users to move between different screens (or pages) in the app. Flutter uses the Navigator widget to handle navigation between routes (screens).

Types of Navigation

- Push Navigation (Forward Navigation) → Moves to a new screen.
- Pop Navigation (Backward Navigation) → Moves back to the previous screen.
- PushReplacement → Replaces the current screen with a new one.
- PushAndRemoveUntil → Moves to a new screen and removes previous screens from the stack.

Routing in Flutter manages different screens in the app. It helps organize and structure navigation efficiently.

Types of Routing

1. Direct Route Navigation (MaterialPageRoute)-Used for simple page-topage navigation.
2. Named Routes (Predefined Routes in main.dart)-Defined in the MaterialApp widget and used throughout the app.

Flutter uses the GestureDetector widget to detect user interactions like taps, swipes, pinches, and long presses. This is essential for making an app interactive.

Common Gestures & Their Uses:

- Tap → Detects simple taps on a widget.
- Double Tap → Recognizes double-clicking.
- Long Press → Triggers an action when the user presses and holds.
- Swipe (Drag) → Detects horizontal or vertical dragging.
- Pinch (Zoom In/Out) → Detects two-finger pinch for zooming.

SYNTAX

Navigator

```
Navigator.push(
context,
  MaterialPageRoute(builder: (context) => SecondPage()),
);
Navigator.pushReplacement(
  context,
  MaterialPageRoute(builder: (context) => NewPage()),
);
```

Routing

```
void main() {
runApp(MaterialApp(
   initialRoute: '/',
routes: {
     '/': (context) => HomePage(),
     '/profile': (context) => ProfilePage(),
   },
  ));
}
```

Gestures

```
GestureDetector(   onTap:
() {     print("Widget
Tapped!");
  },
  child:      Container(
width: 100,     height:
100,                color:
Colors.blue,
  ),
);
```

Widget Properties Navigator context → The current
build context for navigation.
MaterialPageRoute → Creates a transition animation between pages.
builder → Defines the widget to navigate to.
Navigator.push() → Pushes a new screen on top of the stack.
Navigator.pop() → Removes the top screen and goes back.
Navigator.pushReplacement() → Replaces the current screen with a new one.

Routing initialRoute → Sets the first screen when the
app starts.
routes → Defines a map of route names and corresponding widgets.
Navigator.pushNamed() → Navigates using a predefined route.
Navigator.pop() → Closes the current screen and returns to the previous one.
Gestures onDoubleTap → Detects a double tap.

onLongPress → Detects when the user presses and holds.

onHorizontalDragStart → Detects when a horizontal drag begins.

onHorizontalDragUpdate → Detects movement during a horizontal drag.

onHorizontalDragEnd → Detects when a horizontal drag stops.

CODE

```
      // Navigate to Home Page
Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => const TwitterHomePage()),
      );
    }
```

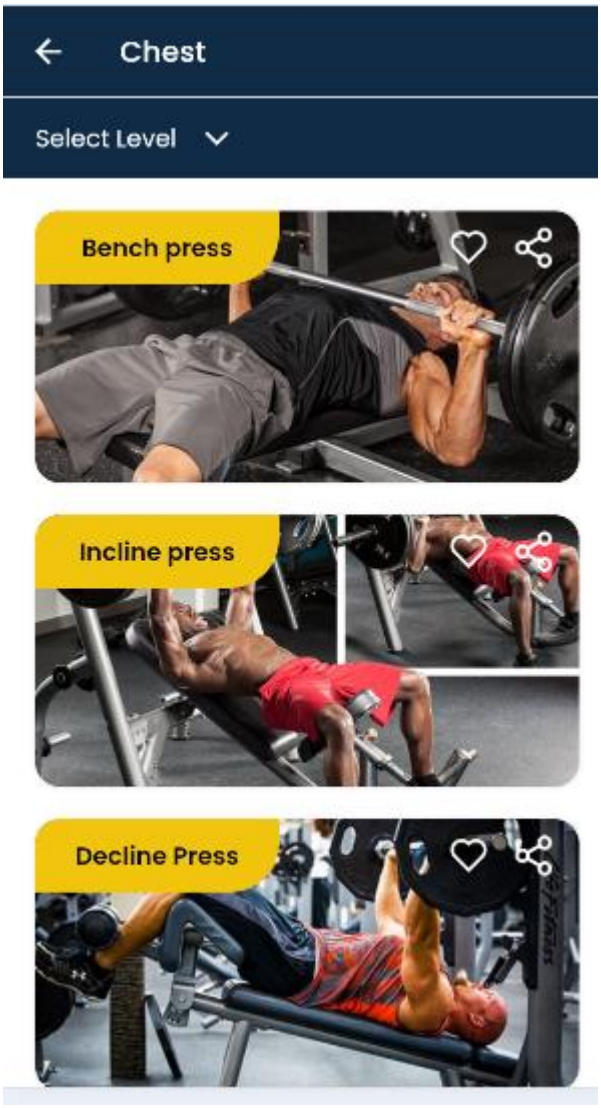
OUTPUT

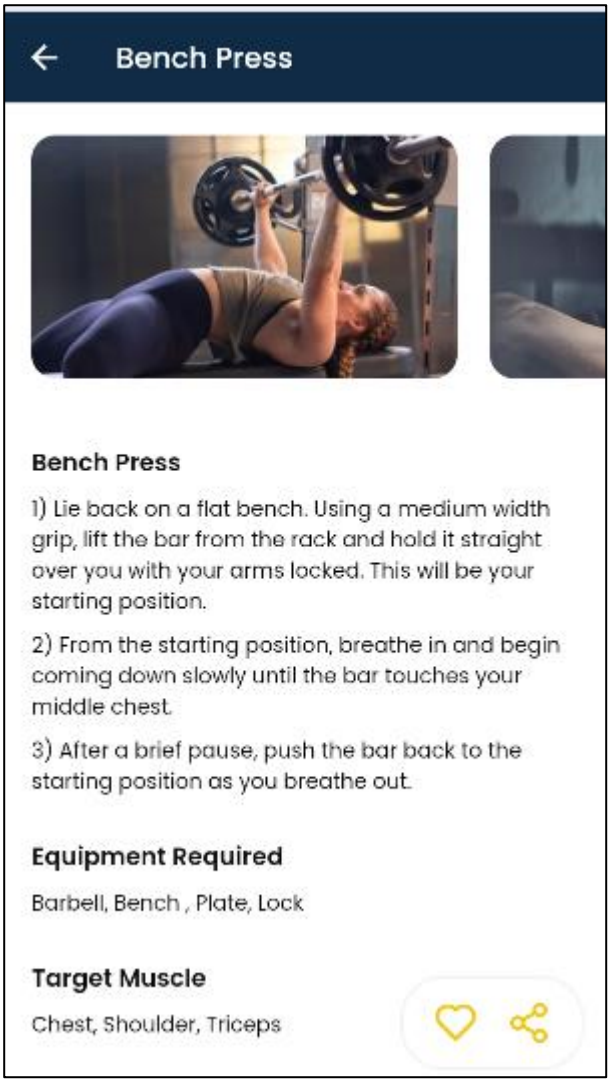
CODE

To go the bench press exercise:

```
 onTap:() {
            Navigator.push(
context,
             MaterialPageRoute(builder: (context) => const UserProfilePage()),
               );
             },
```

OUTPUT


Chest
Select Level
Bench press
Incline press
Decline Press

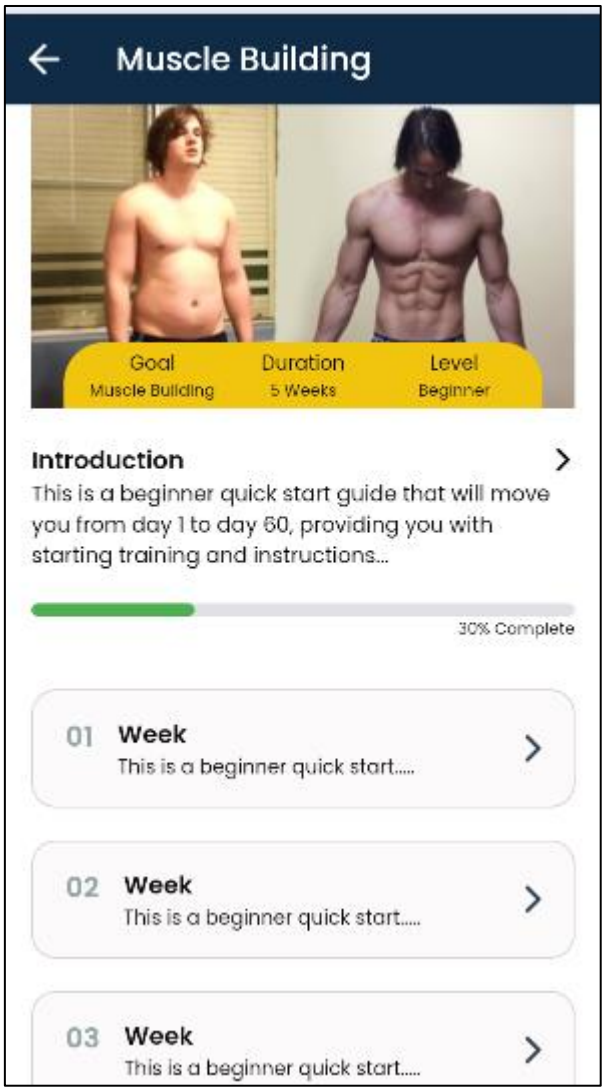

Bench Press
Bench Press
1) Lie back on a flat bench. Using a medium width grip, lift the bar from the rack and hold it straight over you with your arms locked. This will be your starting position.
2) From the starting position, breathe in and begin coming down slowly until the bar touches your middle chest.
3) After a brief pause, push the bar back to the starting position as you breathe out.
Equipment Required
Barbell, Bench , Plate, Lock
Target Muscle
Chest, Shoulder, Triceps

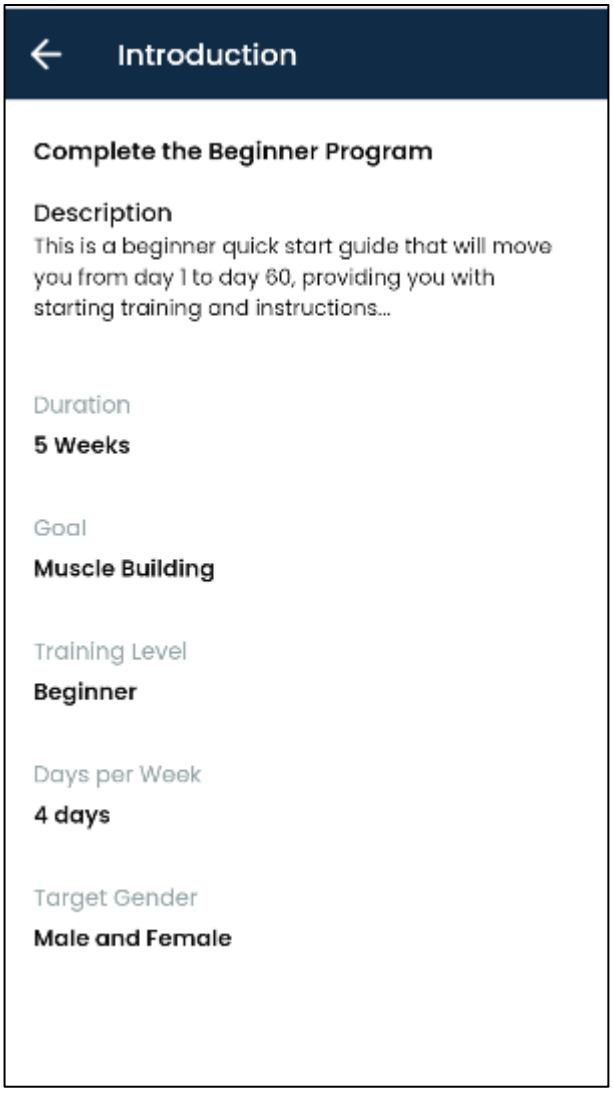
When clicked on introduction of widget

CODE

```
onTap: () {
            Navigator.pushReplacement(
              context,
               MaterialPageRoute(builder: (context) => const TwitterLoginPage())
               );
           },
```

OUTPUT:





MAD & PWA Lab

Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	

EXPERIMENT 7

Name- Rohan Lalchandani

Class-D15A

Roll no- 25

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 end-users to go to the App Store or other such platforms to download the update and wait until installed.

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

6. Discoverable

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

7. Lower Development Cost

Progressive web apps can be installed on the user device like a native device, but it does not demand submission on an App Store. This makes it far more cost-

effective than native mobile applications while offering the same set of functionalities.

Pros and cons of the Progressive Web App

The main features are:

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

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

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

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

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

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

Reactivable — Make it easy to reactivate the application thanks to capabilities such as web notifications.
Installable — They allow the user to “save” the apps that he considers most useful with the corresponding icon on the screen of his mobile terminal (home screen) without having to face all the steps and problems related to the use of the app store.

Linkable — Easily shared via URL without complex installations.

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

Weaknesses refer to:
IOS support from version 11.3 onwards;
Greater use of the device battery;
Not all devices support the full range of PWA features (same speech for iOS and Android operating systems);
It is not possible to establish a strong re-engagement for iOS users (URL scheme, standard web notifications);
Support for offline execution is however limited;
Lack of presence on the stores (there is no possibility to acquire traffic from that channel);
There is no “body” of control (like the stores) and an approval process;
Limited access to some hardware components of the devices;
Little flexibility regarding “special” content for users (eg loyalty programs, loyalty, etc.).

Code: -

```
//Manifest.json:
{
 “short_name”: “Movie Rec”,
 “name”: “Movie Recommendation App”,
 “icons”: [
   {
    “src”: “favicon.ico”,
    “sizes”: “64x64 32x32 24x24 16x16”,
    “type”: “image/x-icon”
   },
   {
    “src”: “logo192.png”,
    “type”: “image/png”,
    “sizes”: “192x192”,
    “purpose”: “any maskable”
   },
   {
    “src”: “logo512.png”,
    “type”: “image/png”,
    “sizes”: “512x512”
   }
 ],
 “start_url”: “.”,
```

```
  “display”: “standalone”,
  “theme_color”: “#000000”,
  “background_color”: “#ffffff”,
  “description”: “Find your next favorite movie”,
  “orientation”: “portrait-primary”,
  “categories”: [“entertainment”, “movies”]
}

//Serviceworker.js

// Import workbox from CDN (you can also use the workbox-webpack-plugin)
importScripts('https://storage.googleapis.com/workboxc
dn/releases/6.4.1/workbox-sw.js'); // Precache and route
setup workbox.routing.registerRoute(
  ({request}) => request.destination === 'image',
new workbox.strategies.CacheFirst({     cacheName:
'images',     plugins: [       new
workbox.expiration.ExpirationPlugin({
maxEntries: 60,         maxAgeSeconds: 30 * 24 * 60
* 60, // 30 Days
      }),
    ],
  })
);

// Cache CSS and JavaScript Files workbox.routing.registerRoute(
  ({request}) => request.destination === 'script' || request.destination === 'style',
new workbox.strategies.StaleWhileRevalidate({
    cacheName: 'static-resources',
```

```
  })
);
//index.html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta       name="description"       content="Movie recommendations
app - find your next favorite film"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>Movie Recommendations</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

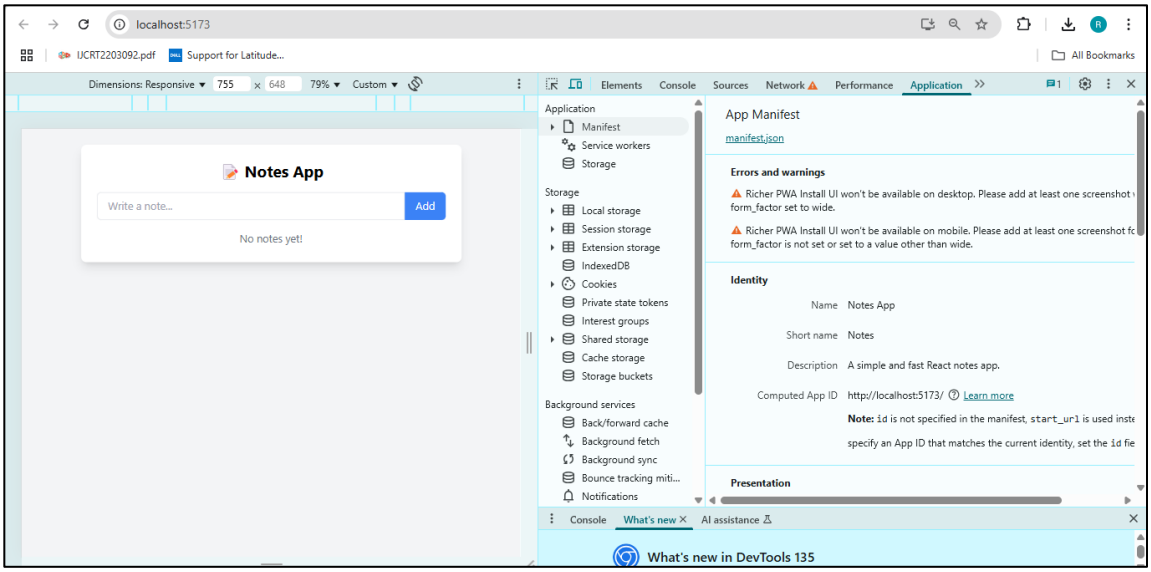
Output

//Open folder in VS Code and run the website

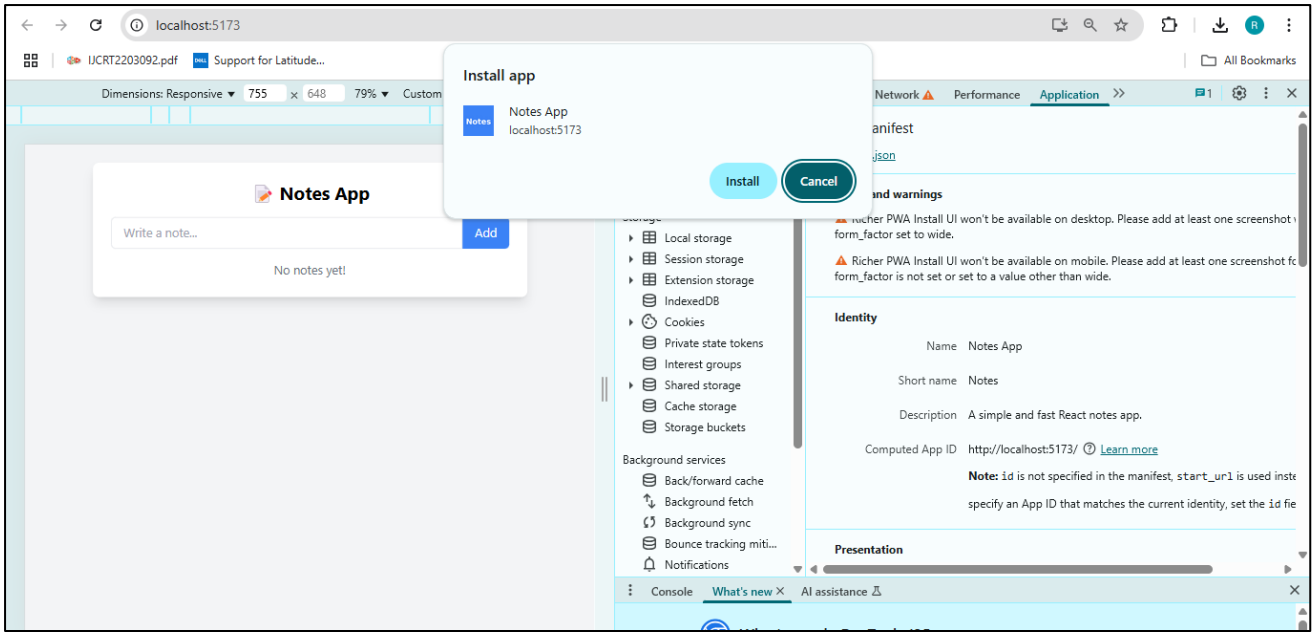
```
HOME@LAPTOP-9JIMM8I3 MINGW64 /e/New folder/Movie-Recommendation-App (main)
$ npm start

Compiled successfully!

You can now view movie-recommendation-app in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.162.1:3000
```

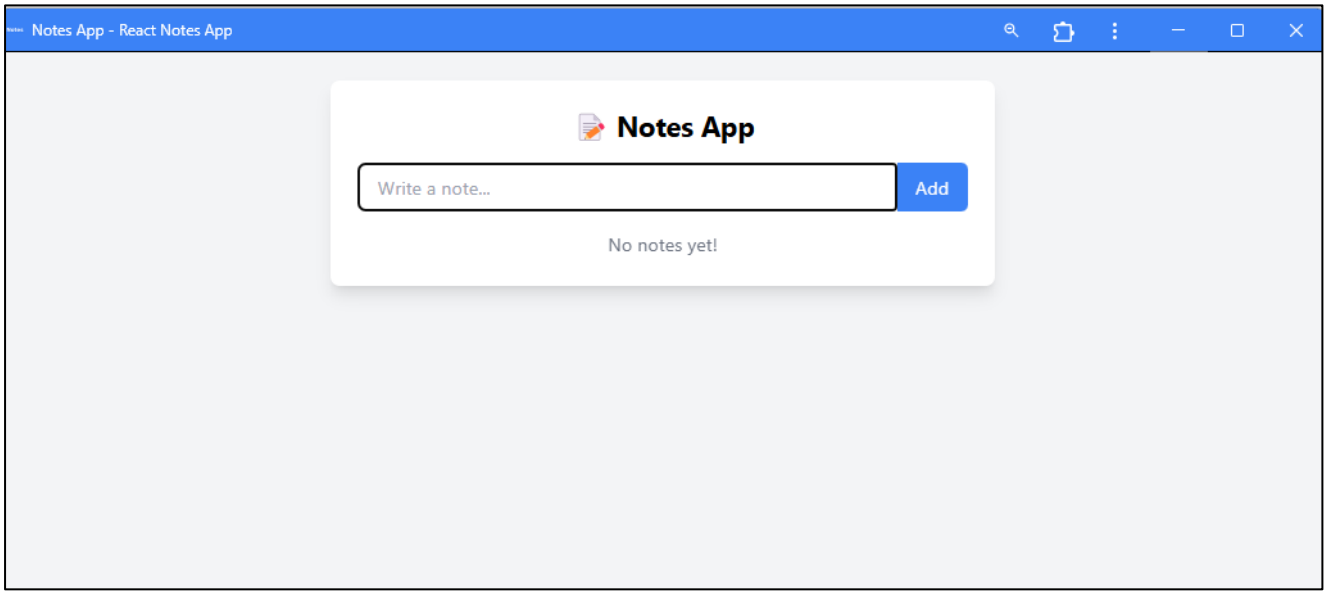
//open developers tools -> Applications

// Install the app by following this route Click on three dots on top right corner of the browser from app option ->Cast,share,save->Install the app



// This is the app





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.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 8

Name- Rohan Lalchandani
Class- D15A
Roll No- 25

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

Theory:

Service Worker

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

Things to note about Service Worker:

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

What can we do with Service Workers?

- You can dominate Network Traffic

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

- You can Cache

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

- You can manage Push Notifications

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

- You can Continue

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

What can’t we do with Service Workers?

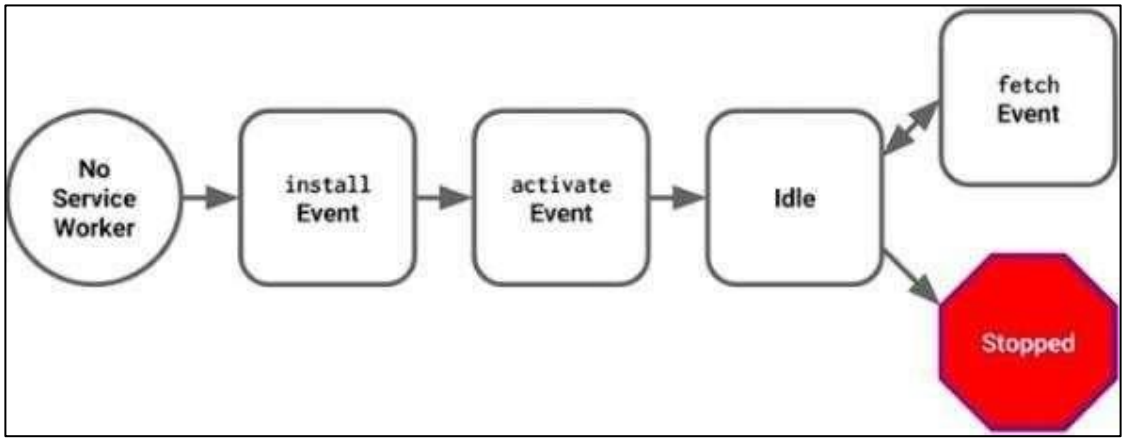
You can’t access the Window

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

- You can’t work it on 80 Port

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

Service Worker Cycle



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

- Registration
- Installation
- Activation

Registration

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

main.js

```
if ('serviceWorker' in navigator) {
navigator.serviceWorker.register('/serviceworker.js')
.then(function(registration) { console.log('Registration
successful, scope is:', registration.scope);
})
.catch(function(error) { console.log('Service worker
registration failed, error:', error);
});
}
```

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

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

You can also set an arbitrary scope by passing in an additional parameter when registering. For example: main.js

```
navigator.serviceWorker.register('/service-worker.js', { scope:
'/app/' });
```

In this case we are setting the scope of the service worker to /app/, which means the service worker will control requests from pages like /app/,

/app/lower/ and /app/lower/lower, but not from pages like /app or /, which are higher.

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

});

Installation

Once the browser registers a service worker, installation can be attempted. This occurs if the service worker is considered to be new by the browser, either because the site currently doesn't have a registered service worker, or because there is a byte difference between the new service worker and the previously installed one.

A service worker installation triggers an install event in the installing service worker. We can include an install event listener in the service worker to perform some task when the service worker installs. For instance, during the install, service workers can precache parts of a web app so that it loads instantly the next time a user opens it (see caching the application shell). So, after that first load, you're going to benefit from instant repeat loads and your time to interactivity is going to be even better in those cases. An example of an installation event listener looks like this:

service-worker.js

// Listen for install event, set callback

self.addEventListener('install', function(event) {

// Perform some task

});

Activation

Once a service worker has successfully installed, it transitions into the activation stage. If there are any open pages controlled by the previous service worker, the new service worker enters a waiting state. The new service worker only activates when there are no longer any pages loaded that are still using the old service worker. This ensures that only one version of the service worker is running at any given time.

When the new service worker activates, an activate event is triggered in the activating service worker. This event listener is a good place to clean up outdated caches (see the Offline Cookbook for an example).

service-worker.js

self.addEventListener('activate', function(event) {// Perform some task });

Once activated, the service worker controls all pages that load within its scope, and starts listening for events from those pages. However, pages in your app that were loaded before the service worker activation will not be under service worker control. The new service worker will only take over when you close and reopen your app, or if the service worker calls clients.claim(). Until then, requests from this page will not be intercepted by the new service worker. This is intentional as a way to ensure consistency in your site.

CODE-

Index.html

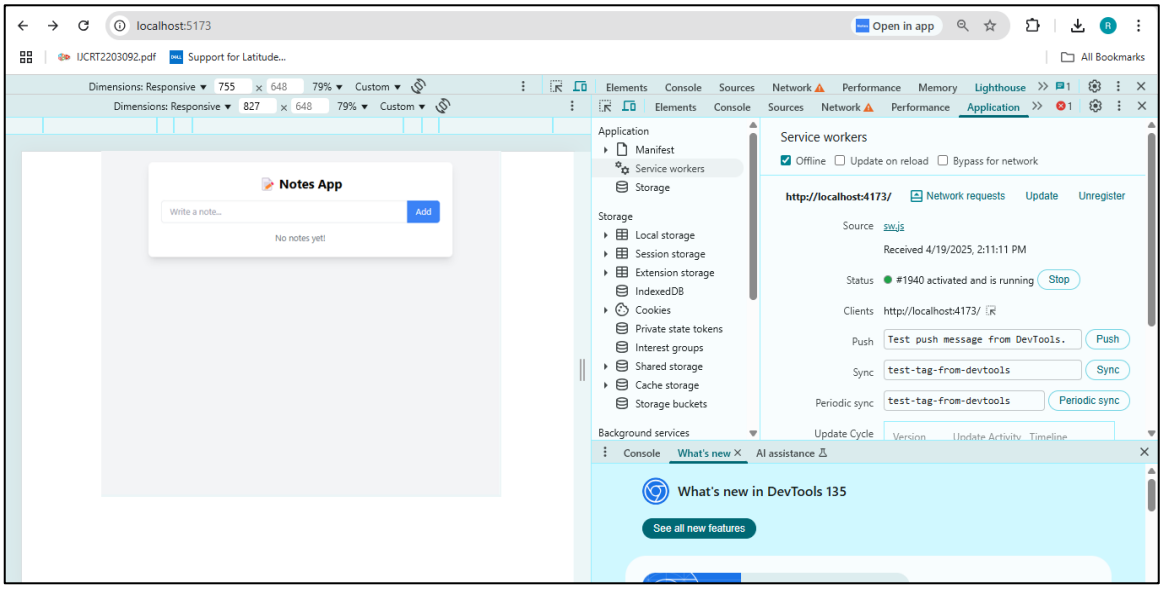
```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <meta name="theme-color" content="#000000" />
    <meta
      name="description"        content="Movie recommendations app
- find your next favorite film"
    />
    <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
    <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
    <title>Movie Recommendations</title>
  </head>
  <body>
    <noscript>You need to enable JavaScript to run this app.</noscript>
    <div id="root"></div>
  </body>
</html>
```

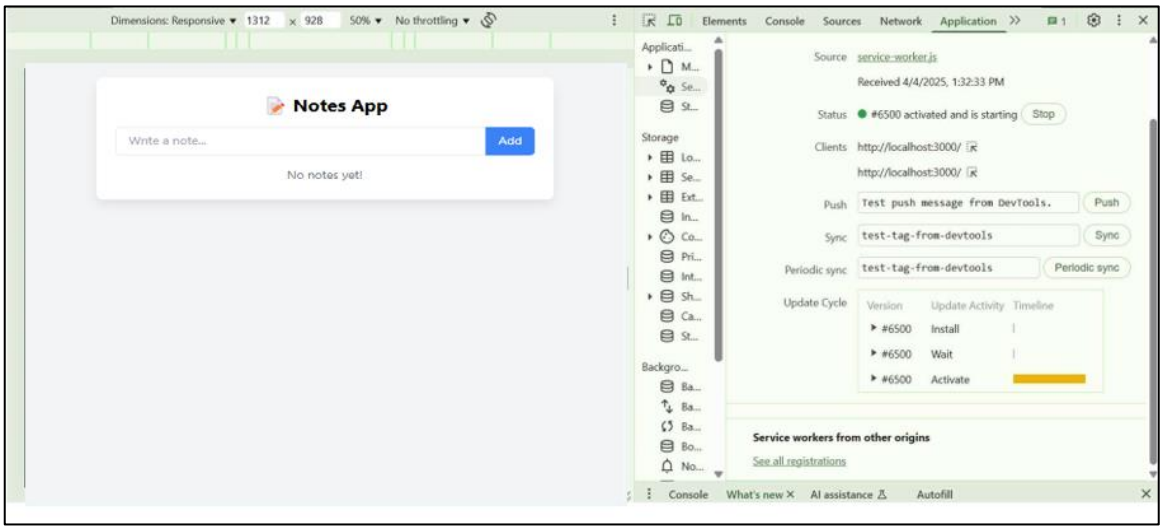
service-worker.js

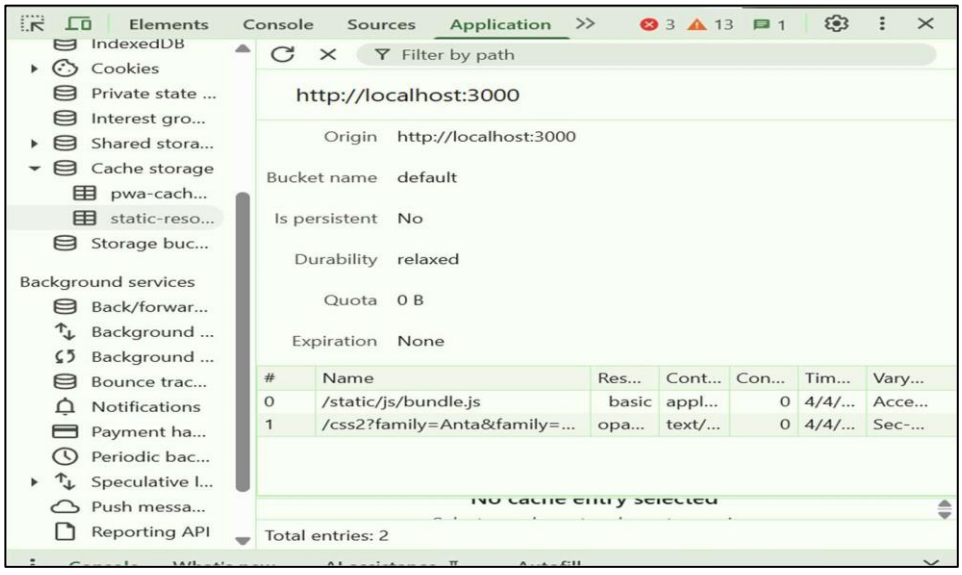
```
importScripts('https://storage.googleapis.com/workboxcdn/releases/6.4.1/workbox-sw.js');

if (workbox) {
  console.log('Workbox loaded');

  const CACHE_NAME = 'pwa-cache-v1';
const PRECACHE_ASSETS = [
    '/',
    '/index.html',
    '/favicon.ico',
    '/logo192.png',
    '/logo512.png',
    '/manifest.json',
    '/static/css/main.css',
    '/static/js/main.js'
  ];

  // Precache assets manually
  workbox.precaching.precacheAndRoute(PRECACHE_ASSETS);

  // Cache images
workbox.routing.registerRoute(
    ({ request }) => request.destination === 'image',
    new workbox.strategies.CacheFirst({
cacheName: 'images-cache',
      plugins: [
        new workbox.expiration.ExpirationPlugin({
maxEntries: 60,
          maxAgeSeconds: 30 * 24 * 60 * 60, // 30 Days
        }),
      ],
    })
  );

  // Cache CSS & JS
  workbox.routing.registerRoute(
    ({ request }) => request.destination === 'script' || request.destination === 'style',
    new workbox.strategies.StaleWhileRevalidate({
cacheName: 'static-resources',
    })
```

```
  );

  // Debug Fetch Requests   self.addEventListener('fetch',
(event) => {     console.log('[Service Worker] Fetching:',
event.request.url);     event.respondWith(
caches.match(event.request).then((cachedResponse) =>
{         return cachedResponse || fetch(event.request);
      })
    );
  });

  // Install Event
  self.addEventListener('install', (event) => {
console.log('[Service Worker] Installing...');
event.waitUntil(
caches.open(CACHE_NAME).then((cache) => {
console.log('[Service Worker] Caching assets');
return cache.addAll(PRECACHE_ASSETS);
      })
    );
    self.skipWaiting();
  });

  // Activate Event
self.addEventListener('fetch', (event) => {
  event.respondWith(

caches.match(event.request).then((response)
=> {
     return response ||
fetch(event.request).catch(() => {
       console.warn('⚠ Failed to fetch:',
event.request.url);
       return new Response('Offline and
resource not cached', {
         status: 503,
         statusText: 'Service Unavailable'
       });
     });
   })
  );
});
```

OUTPUT:


localhost:5173
Notes App
Write a note...
Add
No notes yet!
Service workers
Offline
Update on reload
Bypass for network
http://localhost:4173/
Network requests
Update
Unregister
Source sw.js
Received 4/19/2025, 2:11:11 PM
Status #1940 activated and is running
Stop
Clients http://localhost:4173/
Push Test push message from DevTools.
Push
Sync test-tag-from-devtools
Sync
Periodic sync test-tag-from-devtools
Periodic sync
Update Cycle
What's new in DevTools 135
See all new features


Notes App
Write a note...
Add
No notes yet!
Source service-worker.js
Received 4/4/2025, 1:32:33 PM
Status #6500 activated and is starting
Stop
Clients http://localhost:3000/
http://localhost:3000/
Push Test push message from DevTools.
Push
Sync test-tag-from-devtools
Sync
Periodic sync test-tag-from-devtools
Periodic sync
Update Cycle
Version
Update Activity
Timeline
#6500 Install
#6500 Wait
#6500 Activate
Service workers from other origins
See all registrations


Elements
Console
Sources
Application
IndexedDB
Cookies
Private state ...
Interest gro...
Shared stora...
Cache storage
pwa-cach...
static-reso...
Storage buc...
Background services
Back/forwar...
Background ...
Background ...
Bounce trac...
Notifications
Payment ha...
Periodic bac...
Speculative l...
Push messa...
Reporting API
Filter by path
http://localhost:3000
Origin http://localhost:3000
Bucket name default
Is persistent No
Durability relaxed
Quota 0 B
Expiration None
Name Res... Cont... Con... Tim... Vary...
0 /static/js/bundle.js basic appl... 0 4/4/... Acce...
1 /css2?family=Anta&family=... opa... text/... 0 4/4/... Sec-...
Total entries: 2

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.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 8

Name- Rohan Lalchandani

Class- D15A

Roll No- 25

Aim: To implement Service worker events like fetch, sync and push for Ecommerce PWA

Theory:

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

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

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

Of course, you can use many different methods but you can find in the following example a “cache first” and “network first” approach. In this example, if the request’s and current location’s origin are the same (Static content is requested.), this is called “cacheFirst” but if you request a targeted external URL, this is called “networkFirst”.

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

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 email 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.

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-

Service-worker.js //

Import Workbox

```
importScripts('https://storage.googleapis.com/workboxcdn/releases/6.4.1/workbox-sw.js');
```

```
if (workbox) {
  console.log(' Workbox Loaded Successfully');

  // Precache assets
workbox.precaching.precacheAndRoute(self.__WB_MANIFEST || []);

  //  Cache Images (Cache-First Strategy)
workbox.routing.registerRoute(
   ({ request }) => request.destination === 'image',
new workbox.strategies.CacheFirst({       cacheName:
'images-cache',       plugins: [         new
workbox.expiration.ExpirationPlugin({
maxEntries: 60,           maxAgeSeconds: 30 * 24 * 60
* 60, // 30 Days
        }),
      ],
   })
  );

  //  Cache CSS & JS (Stale-While-Revalidate Strategy)
workbox.routing.registerRoute(
   ({ request }) => request.destination === 'script' || request.destination ===
'style',
   new workbox.strategies.StaleWhileRevalidate({
cacheName: 'static-resources',
   })
  );
```

```
  // Cache API Responses (Network-First Strategy)
workbox.routing.registerRoute(
   ({ url }) => url.origin.includes('api.themoviedb.org'),
new workbox.strategies.NetworkFirst({
cacheName: 'api-cache',       plugins: [         new
workbox.expiration.ExpirationPlugin({
maxEntries: 50,           maxAgeSeconds: 5 * 60, // 5
minutes
       }),
     ],
   })
  );

}

//  Fetch Event Logging (Works outside Workbox)
self.addEventListener('fetch', (event) => {     console.log(`
Fetch event detected: ${event.request.url}`);

   if (event.request.url.includes('api.themoviedb.org')) {
console.log(`Intercepting API Request: ${event.request.url}`);

     event.respondWith(         caches.match(event.request).then((cachedResponse)
=> {           return cachedResponse || fetch(event.request).then((response) => {
console.log(` API Response Fetched: ${event.request.url}`);             return
```

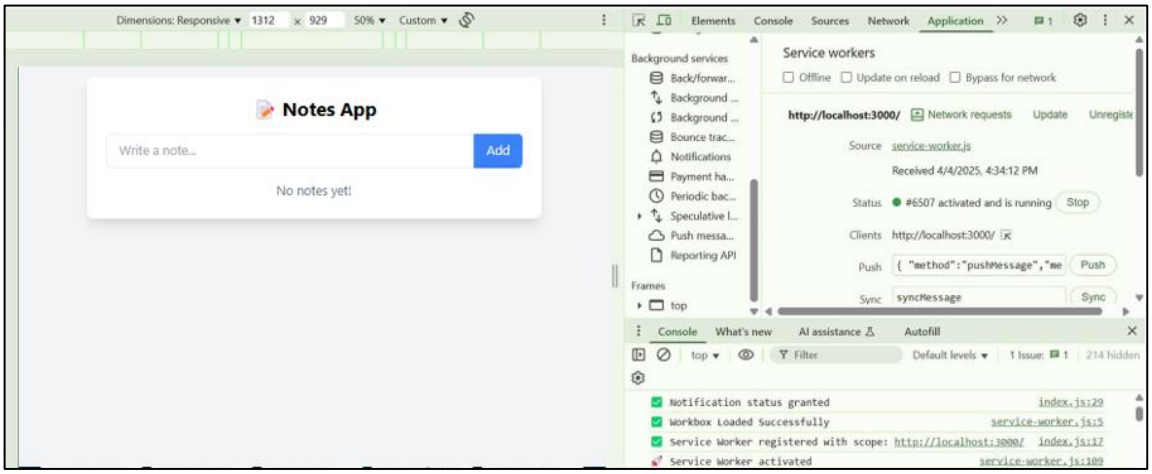
```
response;          }).catch((err) => {             console.error(` API Fetch Failed:
${event.request.url}`, err);             return new Response('API fetch failed', {
status: 500 });
          });
        })
      );
    }
  });

//  Background Sync Event (Syncing Watchlist)
self.addEventListener('sync', (event) => {   if
(event.tag === 'sync-watchlist') {     console.log(' Sync
event triggered: sync-watchlist');     event.waitUntil(
syncWatchlist().then(() => {         console.log(' Sync
successful');
     }).catch((err) => {
console.error(' Sync failed:', err);
     })
   );
  }
});

//  Example Function to Sync Watchlist async
function syncWatchlist() {   console.log(' Syncing
watchlist data...');   return fetch('/sync-watchlist', {
method: 'POST' })
```

```
    .then(() => console.log(' Sync request sent successfully!'))
    .catch(() => console.log(' Sync request failed, retrying later.'));
}

//  Push Notification Event self.addEventListener('push',
(event) => {   console.log('Push notification received');

  const notificationData = event.data ? event.data.text() ;
console.log(`큕 큖 큗 큘 큙 큒 큓 큔 Push payload: ${notificationData}`);

  const options = {
body: notificationData,
icon: '/logo192.png',
badge: '/logo192.png',
  };

  event.waitUntil(     self.registration.showNotification('
Movie House', options)
      .then(() => console.log(' Push notification sent successfully'))
      .catch((err) => console.error(' Push notification failed:', err))    );
});

//  Activate event - Cleanup old caches self.addEventListener('activate',
(event) => {   console.log(' Service Worker activated');   event.waitUntil(
caches.keys().then((cacheNames) =>       Promise.all(
cacheNames.map((cache) => {          if (!['images-cache', 'static-resources',
```

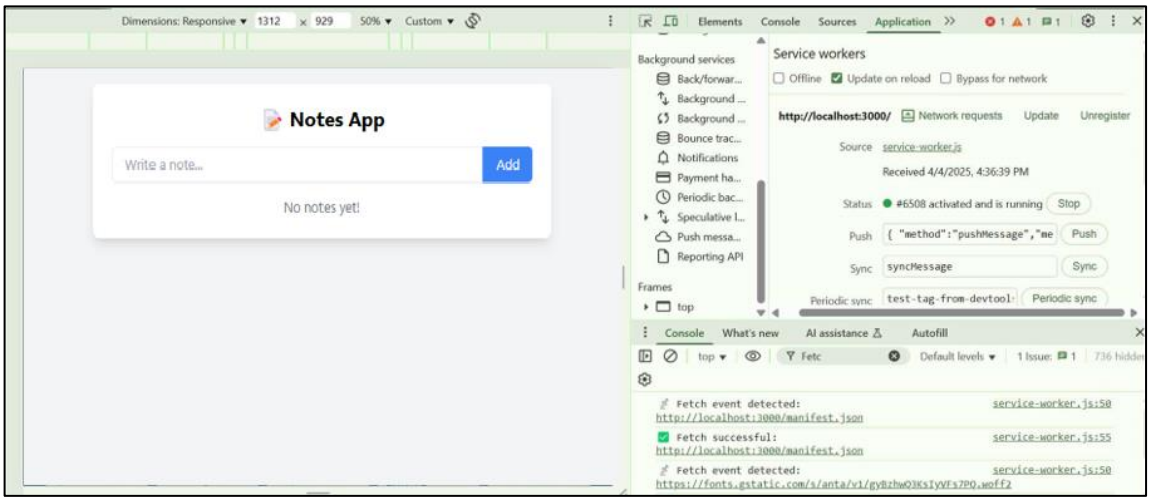
```
'api-cache'].includes(cache)) {            console.log('🗑 Deleting old cache:',
cache);            return caches.delete(cache);
        }
      })
    )
  )
 );
 self.clients.claim();
});
```

OUTPUT

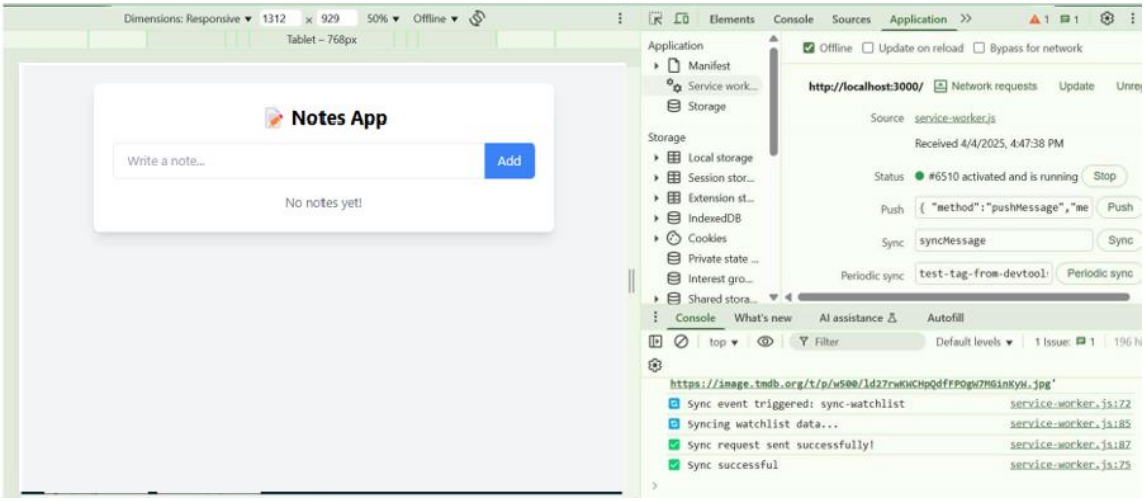
Service worker activated



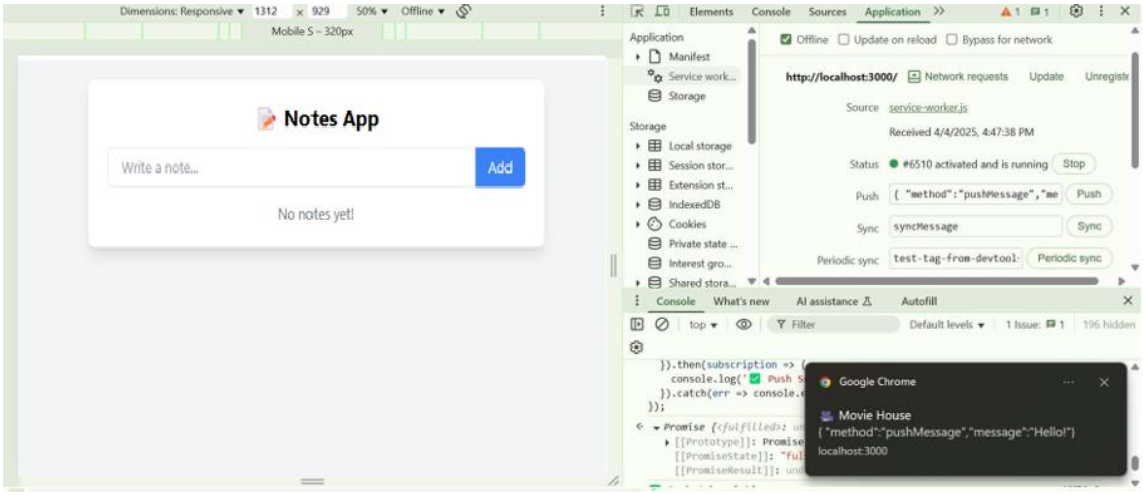
Fetch Event



Sync Event



Push Event


Notes App
Write a note...
Add
No notes yet!
Application
Manifest
Service work...
Storage
Local storage
Session stor...
Extension st...
IndexedDB
Cookies
Private state ...
Interest gro...
Shared stora...
Offline
Update on reload
Bypass for network
http://localhost:3000/
Network requests
Source
service-worker.js
Received 4/4/2025, 4:47:38 PM
Status
#6510 activated and is running
Stop
Push
Sync
syncMessage
Periodic sync
test-tag-from-devtool
Periodic sync
Console
What's new
AI assistance
Autofill
Google Chrome
Movie House
{"method":"pushMessage","message":"Hello!"}
localhost:3000

MAD & PWA Lab

Journal

Experiment No.	10
Experiment Title.	To study and implement deployment of Ecommerce PWA to GitHub Pages.
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	

Experiment No. 8

Name- Rohan Lalchandani
Class- D15A
Roll No- 25

Aim:

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

Theory:

GitHub Pages

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

GitHub Pages provides the following key features:

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

Reasons for favoring this over Firebase:

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

GitHub Pages is used by Lyft, CircleCI, and HubSpot.
GitHub Pages is listed in 775 company stacks and 4401 developer stacks.

Pros

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

Cons

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

Firebase

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

Some of the features offered by Firebase are:

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

Reasons for favoring over GitHub Pages:

1. Realtime backend made easy
2. Fast and responsive

Instacart, 9GAG, and Twitch are some of the popular companies that use Firebase

Firebase has a broader approval, being mentioned in 1215 company stacks & 4651 developers stacks

Pros

1. Hosted by Google. Enough said.

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

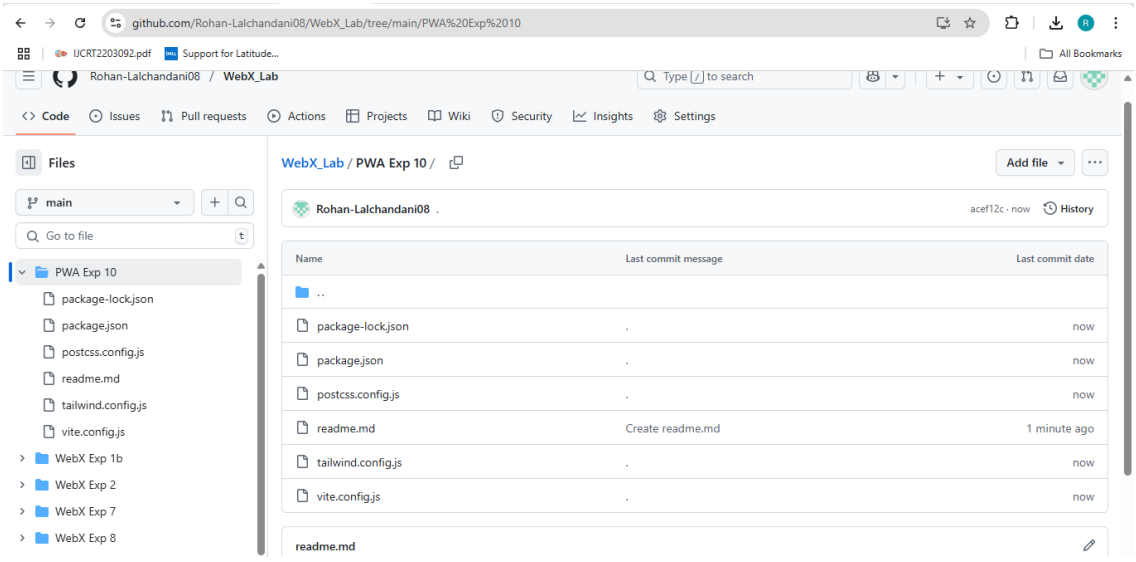
Cons

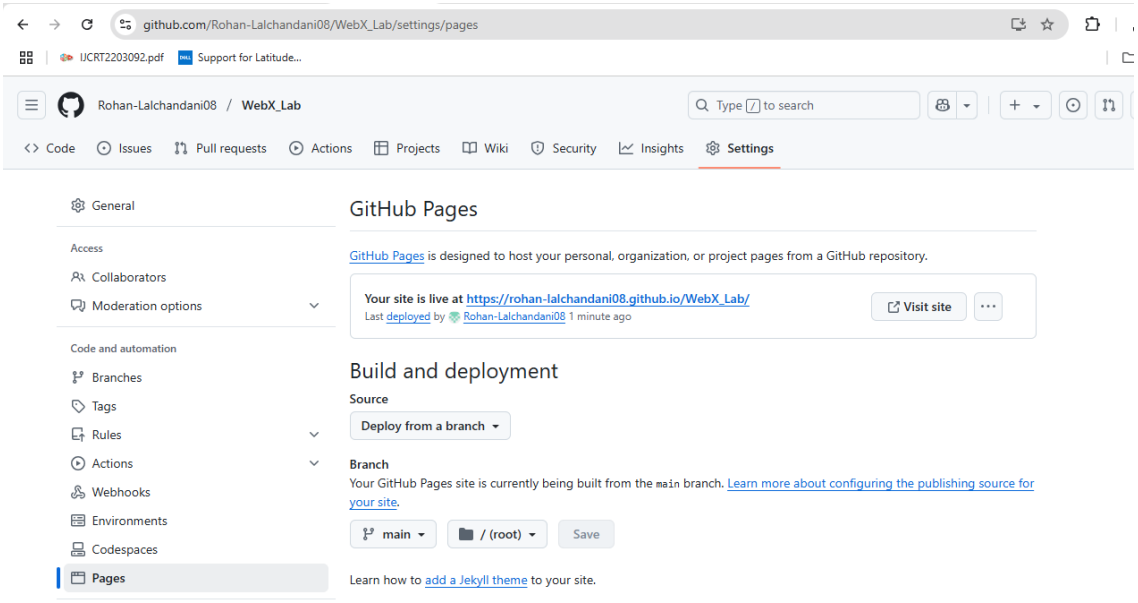
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.
3. No in-built support for any static site generator.

Link to Github Repository-

https://github.com/Rohan-Lalchandani08/WebX_Lab/tree/main/PWA%20Exp%2010

OUTPUT-




github.com/Rohan-Lalchandani08/WebX_Lab/settings/pages
IJCRT2203092.pdf
Support for Latitude...
Rohan-Lalchandani08 / WebX_Lab
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages
GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
Your site is live at https://rohan-lalchandani08.github.io/WebX_Lab/
Last deployed by Rohan-Lalchandani08 1 minute ago
Visit site
Build and deployment
Source
Deploy from a branch
Branch
Your GitHub Pages site is currently being built from the main branch. Learn more about configuring the publishing source for your site.
main
/ (root)
Save
Learn how to add a Jekyll theme to your site.

MAD & PWA Lab

Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	

Experiment 11

Name- Rohan Lalchandani
Class-D15A
Roll no- 25

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

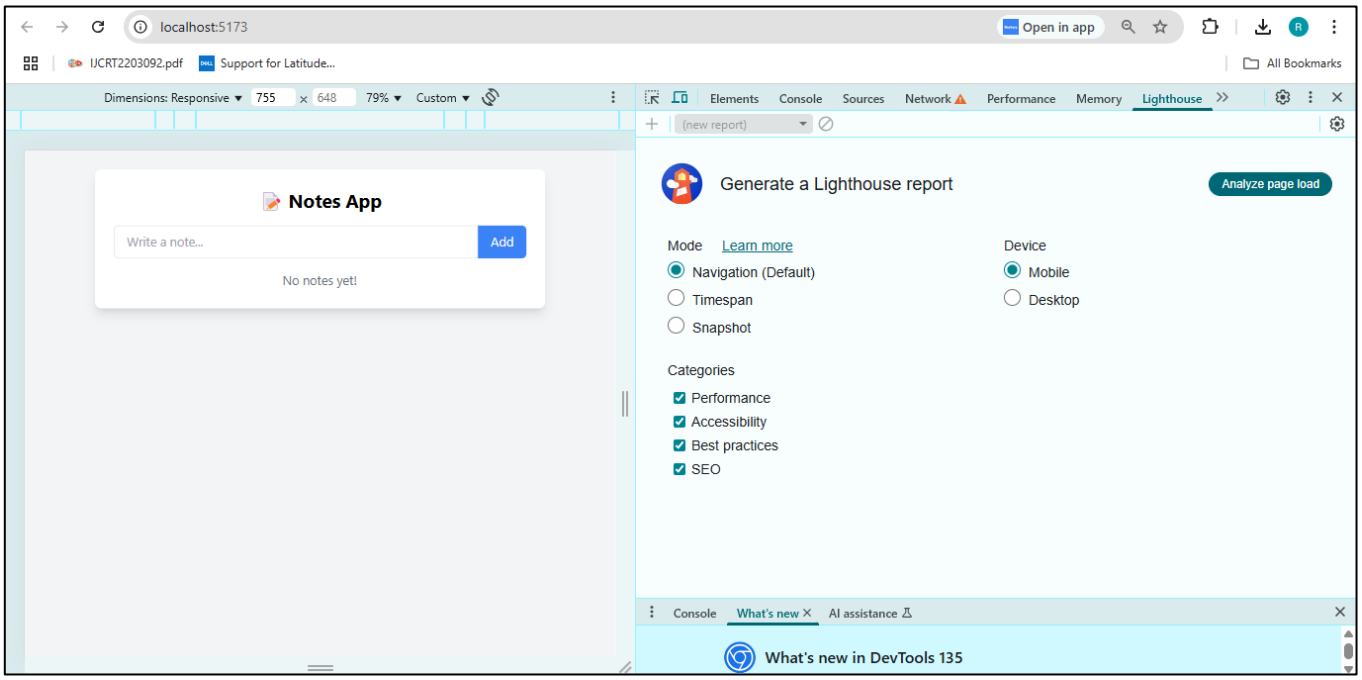
Google Lighthouse has the option of running the Audit for Desktop as well as mobile version of your page(s). The top metrics that will be measured in the Audit are:

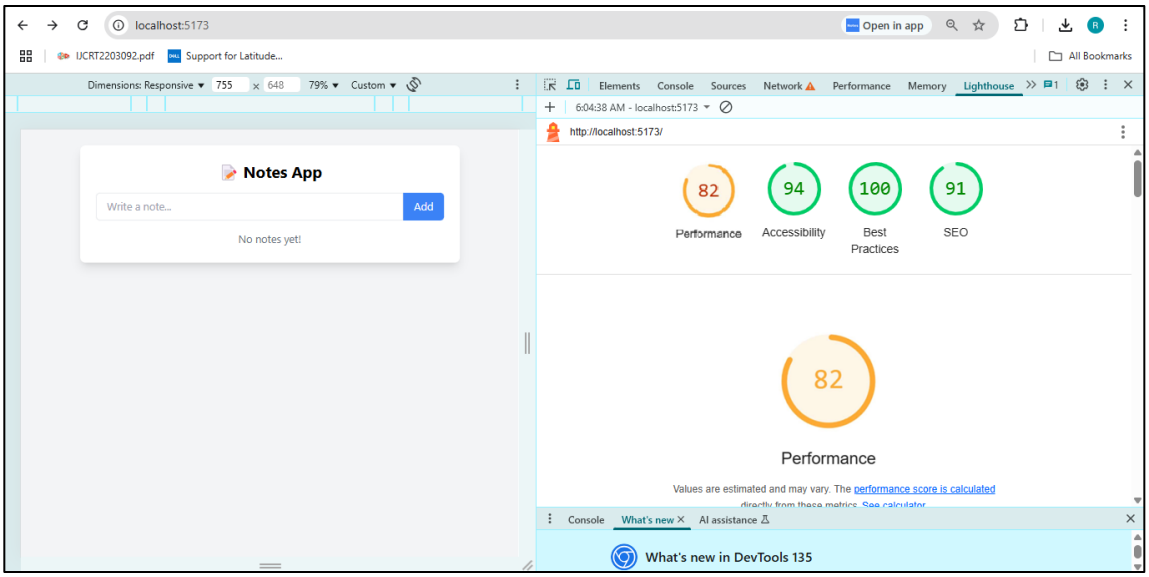
1. Performance: This score is an aggregation of how the page fared in aspects such as (but not limited to) loading speed, time taken for loading for basic frame(s), displaying meaningful content to the user, etc. To a layman, this score is indicative of how decently the site performs, with a score of 100 meaning that you figure in the 98th percentile, 50 meaning that you figure in the 75th percentile and so on.

2. PWA Score (Mobile): Thanks to the rise of Service Workers, app manifests, etc., a lot of modern web applica ons are moving towards the PWA paradigm, where the objec ve is to make the applica on behave as close as possible to na ve mobile applica ons. Scoring points are based on the Baseline PWA checklist laid down by Google which includes Service Worker implementa on(s), viewport handling, offline func onality, performance in script disabled environments, etc.

3. Accessibility: As you might have guessed, this metric is a measure of how accessible your website is, across a plethora of accessibility features that can be implemented in your page (such as the ‘aria-’ a ributes like aria-required, audio cap ons, bu on 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 <sec on>, <ar cle>, etc.), you get a 0 on that score. The aggregate of these scores is your Accessibility metric score. 4. Best Prac ces: As any developer would know, there are a number of prac ces that have been deemed ‘best’ based on empirical data. This metric is an aggrega on of many such points, including but not limited to:Use of HTTPS Avoiding the use of deprecated code elements like tags, direc ves, libraries, etc.
Password input with paste-into disabled
Geo-Loca on and cookie usage alerts on load, etc.

OUTPUT:





CONCLUSION-

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

MAD & PWA Lab

Journal

Experiment No.	Assignment-2
Assignment 2 Questions	1. Define Progressive Web App (PWA) and explain its significance in modern web development.Discuss the key characteristics that differentiate PWAs from traditional mobile apps 2. Define responsive web design and explain its importance in the context of Progressive Web Apps.Compare and contrast responsive, fluid, and adaptive web design approaches. 3. Describe the lifecycle of Service Workers, including registration, installation, and activation phases. 4. Explain the use of IndexedDB in the Service Worker for data storage.
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO4:Understand various PWA frameworks and their requirements LO5: Design and Develop a responsive User Interface by applying PWA Design techniques LO6:Develop and Analyze PWA Features and deploy it over app hosting solutions
Grade:	

MAD & PWA Lab
Journal

Experiment No.	Assignment-1
Assignment 1 Questions	1. Flutter Overview:Explain the key features and advantages of using Flutter for mobile app development. Discuss how the Flutter framework differs from traditional approaches and why it has gained popularity in the developer community. 2. Widget Tree and Composition: Describe the concept of the widget tree in Flutter. Explain how widget composition is used to build complex user interfaces. Provide examples of commonly used widgets and their roles in creating a widget tree. 3. State Management in Flutter: Discuss the importance of state management in Flutter applications. Compare and contrast the different state management approaches available in Flutter, such as setState, Provider, and Riverpod. Provide scenarios where each approach is suitable. 4. Firebase Integration in Flutter: Explain the process of integrating Firebase with a Flutter application. Discuss the benefits of using Firebase as a backend solution. Highlight the Firebase services commonly used in Flutter development and provide a brief overview of how data synchronization is achieved.
Roll No.	25
Name	Rohan Lalchandani
Class	D15A/D15B
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	