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

Lab Assistant

Subject Teacher

Mrs. Kajal Joseph

Principal

Head of Department

Dr. Mrs. Shalu Chopra

Name of the Course : MAD & PWA Lab

Course Code : ITL604

Project Title: Flutter: JioSaavn Clone/ PWA: Fast Food Delivery **Roll No. 10**

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

Faculty Incharge : Mrs. Kajal Joseph.

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

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

Programme Outcomes: The graduate will be able to:

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

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

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

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

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

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

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

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

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

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

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

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

Program specific Outcomes

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

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

Project Title: Flutter: JioSaavn Clone/ PWA: Fast Food Delivery **Roll No. 10**

Lab Objectives:

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

Lab Outcomes:

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

Index

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

MAD & PWA Lab

Journal

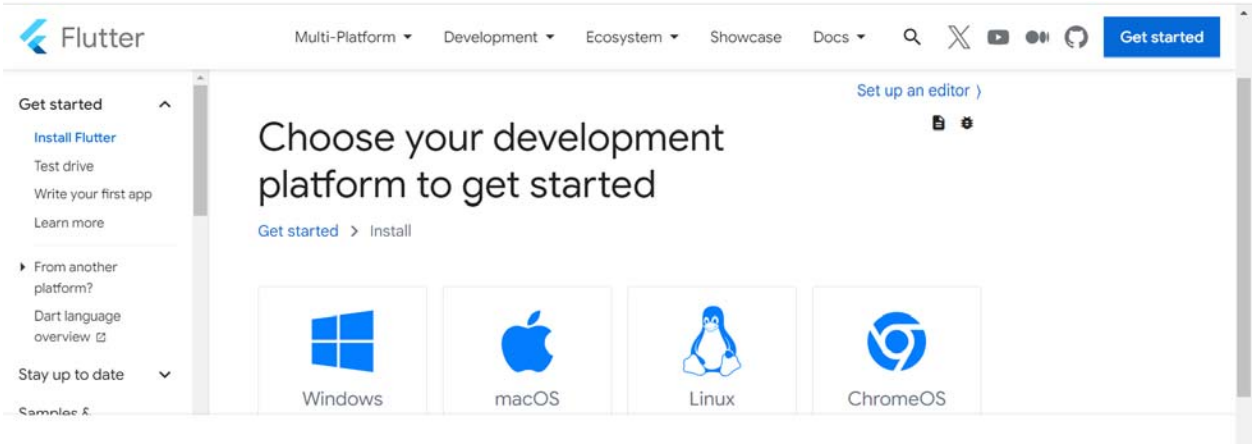
Experiment No.	01
Experiment Title.	To install and configure the Flutter Environment
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10

MAD PWA Exp 1

Aim: Installation and Configuration of Flutter Environment.

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



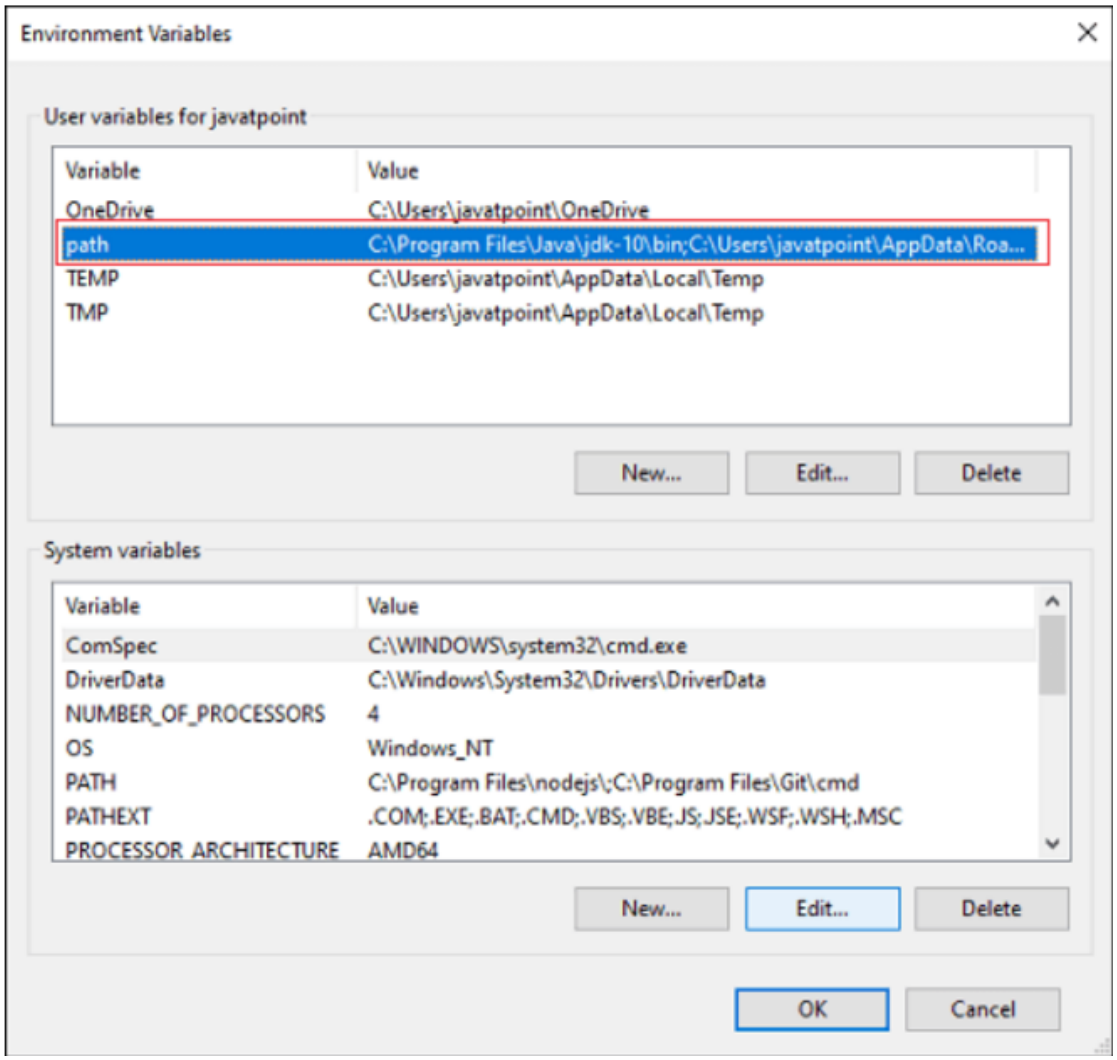
Step 2: Next, to download the latest Flutter SDK, click on the Windows icon. Here, you will find the download link for SDK.

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

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

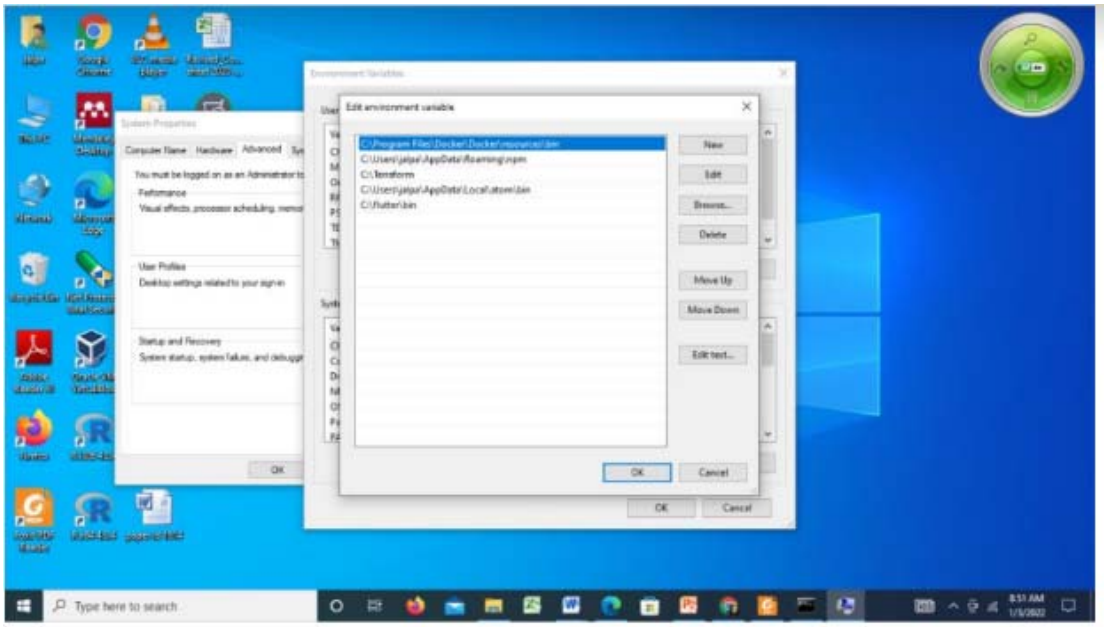
Step 4.1: Go to MyComputer properties -> advanced tab -> environment variables. You will get the following screen.

Name: Atharva Chavan Div: D15A Roll no: 10



Step 4.2: Now, select path -> click on edit. The following screen appears

Name: Atharva Chavan Div: D15A Roll no: 10



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

Step 5: Now, run the $ flutter command in the command prompt. Now, run the $ flutter doctor command. This command checks for all the requirements of Flutter app development and displays a report of the status of your Flutter installation.

Step 6: When you run the above command, it will analyze the system and show its report, as shown in the below image. Here, you will find the details of all missing tools, which are required to run Flutter as well as the development tools that are available but not connected with the device.

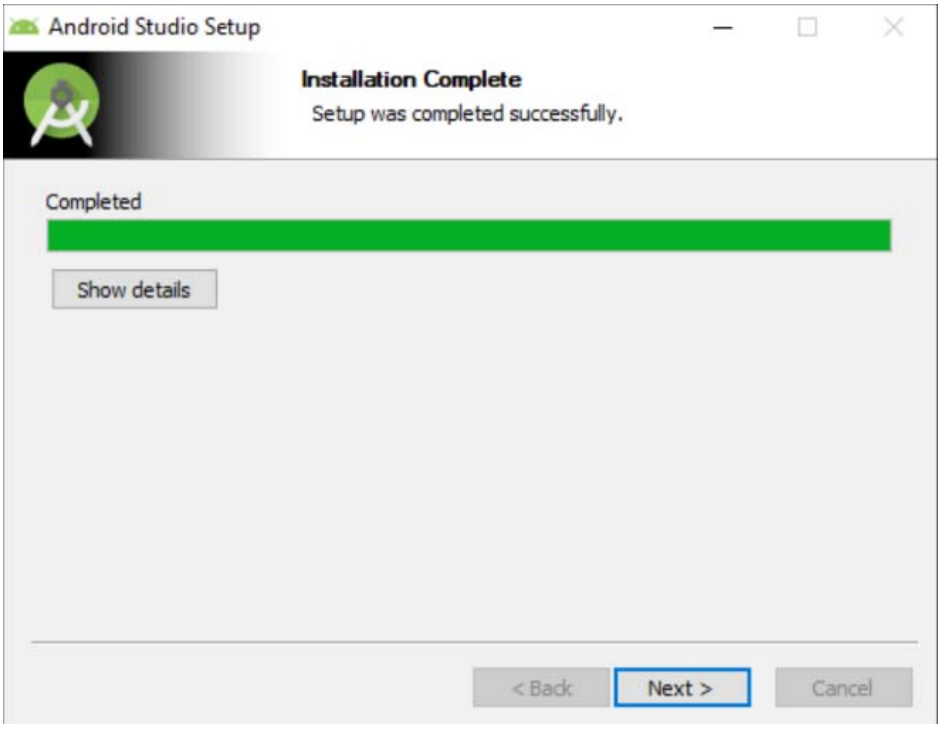
Step 7: Install the Android SDK. If the flutter doctor command does not find the Android SDK tool in your system, then you need first to install the Android Studio IDE. To install Android Studio IDE, do the following steps.

Name: Atharva Chavan Div: D15A Roll no: 10

Step 7.1: Download the latest Android Studio executable or zip file from the official site.

Step 7.2: When the download is complete, open the .exe file and run it. You will get the following dialog box.

Step 7.3: Follow the steps of the installation wizard. Once the installation wizard completes, you will get the following screen.



Step 7.4: In the above screen, click Next-> Finish. Once the Finish button is clicked, you need to choose the 'Don't import Settings option' and click OK. It will start the Android Studio.

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

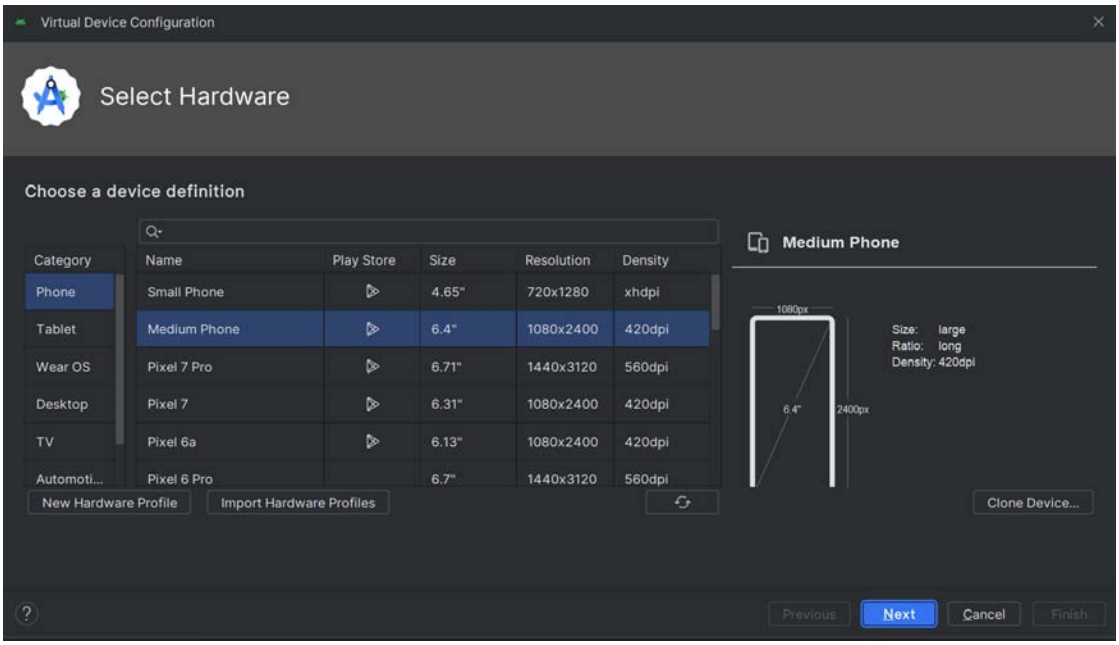
Step 8.1: To set an Android emulator, go to Android Studio > Tools > Android > AVD Manager and select Create Virtual Device. Or, go to Help->Find Action->Type Emulator in the search box. You will get the following screen.

Step 8.2: Choose your device definition and click on Next.

Step 8.3: Select the system image for the latest Android version and click on Next.

Name: Atharva Chavan Div: D15A Roll no: 10

Step 8.4: Now, verify the all AVD configuration. If it is correct, click on Finish. The following screen appears.



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

Step 9: Now, install the Flutter and Dart plugin for building Flutter applications in Android Studio. These plugins provide a template to create a Flutter application, give an option to run and debug Flutter application in the Android Studio itself. Do the following steps to install these plugins.

Step 9.1: Open the Android Studio and then go to File->Settings->Plugins.

Step 9.2: Now, search the Flutter plugin. If found, select Flutter plugin and click install. When you click on install, it will ask you to install Dart plugin as below screen. Click yes to proceed.
Step 9.3: Restart the Android Studio.

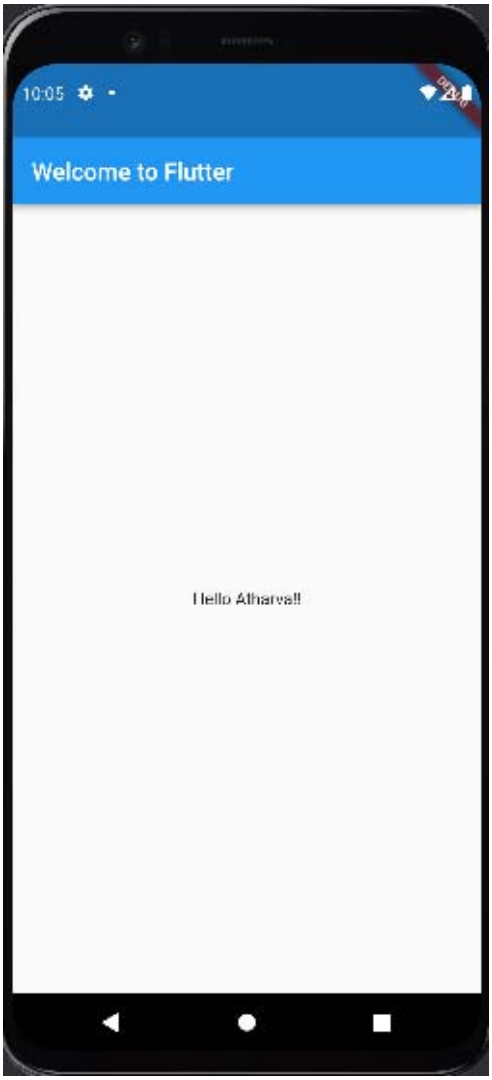
Code:

Name: Atharva Chavan Div: D15A Roll no: 10

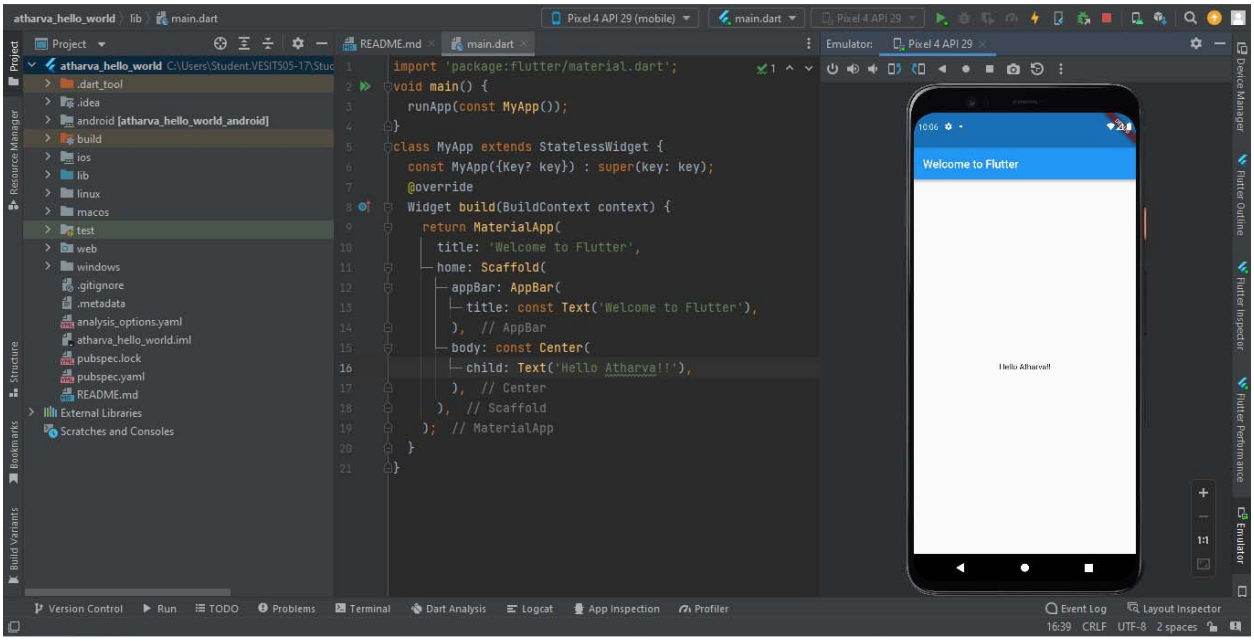
```
import 'package:flutter/material.dart';
void main() {
  runApp(const MyApp());
}
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Welcome to Flutter',
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Welcome to Flutter'),
        ),
        body: const Center(
          child: Text('Hello Atharva!!'),
        ),
      ),
    );
  }
}
```

Output:

Name: Atharva Chavan Div: D15A Roll no: 10



Name: Atharva Chavan Div: D15A Roll no: 10



MAD & PWA Lab

Journal

Experiment No.	02
Experiment Title.	To design Flutter UI by including common widgets.
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10 Batch A

MAD PWA Exp 2

Aim: To design Flutter UI by including common widgets.

Theory:

We can split the Flutter widget into two categories:

1. Visible (Output and Input)
2. Invisible (Layout and Control)

1. Visible widget

The visible widgets are related to the user input and output data. Some of the important types of

this widget are:

1. Text

A Text widget holds some text to display on the screen. We can align the text widget by using

textAlign property, and style property allow the customization of Text that includes font, font

weight, font style, letter spacing, color, and many more.

2. Button

This widget allows you to perform some action on click. Flutter does not allow you to use the

Button widget directly; instead, it uses a type of buttons like a FlatButton and a RaisedButton.

3. Image

This widget holds the image which can fetch it from multiple sources like from the asset

folder or directly from the URL. It provides many constructors for loading image, which

are given below:

o Image: It is a generic image loader, which is used by ImageProvider.

o asset: It load image from your project asset folder.

o file: It loads images from the system folder.

o memory: It load image from memory.

o network: It loads images from the network.

To add an image in the project, you need first to create an assets folder where you keep

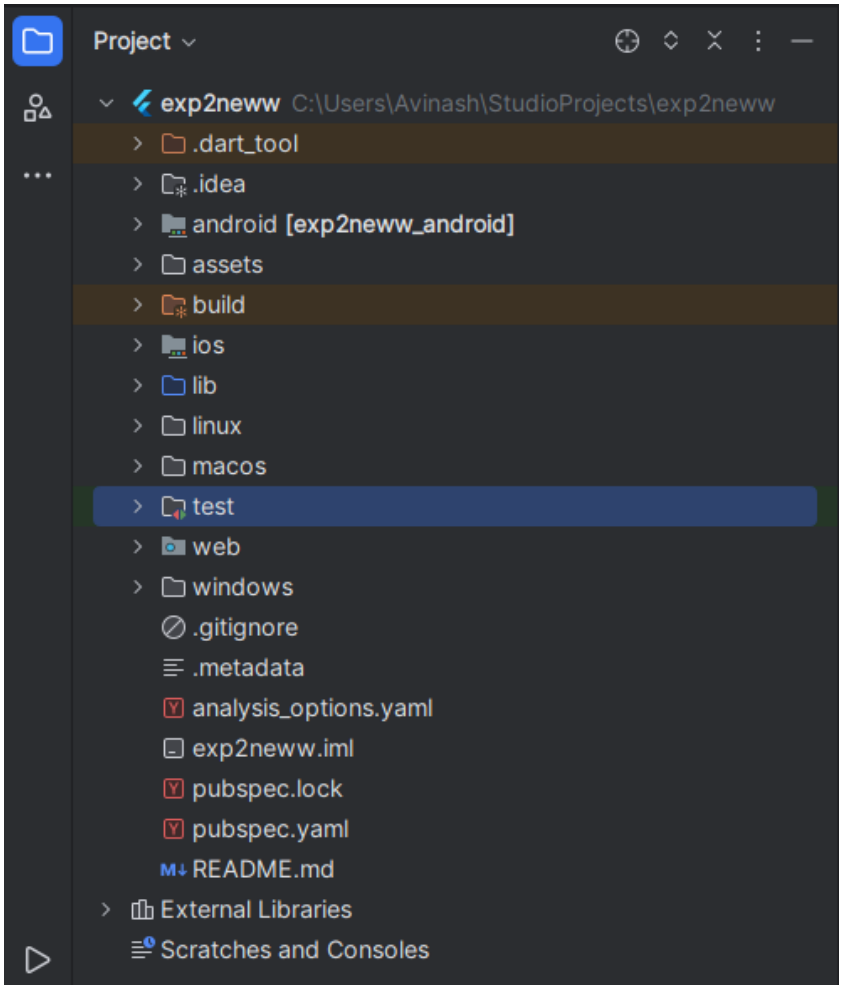
your images and then add the below line in pubspec.yaml file.

assets:

- assets/comp.jpg

Name: Atharva Chavan Div: D15A Roll no: 10 Batch A

File structure:



Code:

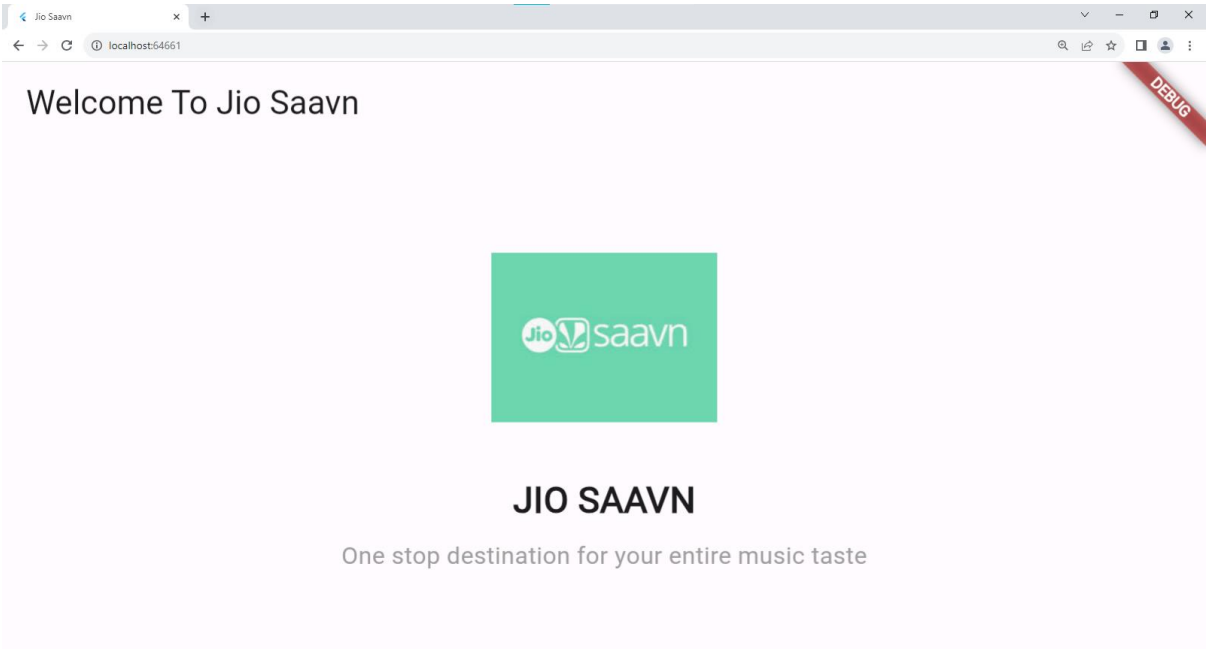
```
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Jio Saavn',
      home: HomeScreen(),
    );
  }
}

class HomeScreen extends StatelessWidget {
```

Name: Atharva Chavan Div: D15A Roll no: 10 Batch A

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Welcome To Jio Saavn'),
        backgroundColor: Colors.transparent,
        elevation: 0,
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Image.asset(
              'assets/jiosaavn.jpg',  // Update with your actual logo
path
              width: 150,  // Adjust the width as needed
              height: 150, // Adjust the height as needed
            ),
            SizedBox(height: 16),
            Text(
              'JIO SAAVN',
              style: TextStyle(
                fontSize: 24,
                fontWeight: FontWeight.bold,
              ),
            ),
            SizedBox(height: 8),
            Text(
              'One stop destination for your entire music taste',
              style: TextStyle(
                fontSize: 16,
                color: Colors.grey,
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

Name: Atharva Chavan Div: D15A Roll no: 10 Batch A

Output:



Conclusion:

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

MAD & PWA Lab

Journal

Experiment No.	03
Experiment Title.	To include icons, images, fonts in Flutter app
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

MAD PWA Exp 3

Aim: To include icons, images and fonts in flutter app

Theory:

Flutter, Google's open-source UI toolkit, empowers developers to craft visually appealing and highly performant applications. A fundamental aspect of creating a compelling user interface lies in the seamless integration of icons, images, and custom fonts.

Icons

Icons serve as intuitive visual cues, aiding users in navigation and comprehension. Flutter provides an extensive set of built-in icons, and developers can easily integrate custom icons. Icons contribute to a consistent design language and enhance the overall user experience.

Principles:

1. Selecting Icon Packages: Developers often leverage existing icon packages, such as "material_icons" or "font_awesome_flutter," to access a diverse range of symbols. This streamlines the integration process and ensures a cohesive visual language.
2. Installation and Implementation: The chosen icon package is added to the project's dependencies. Icons can then be effortlessly implemented using the Icon widget, specifying the desired icon type, size, and color.

Images

Images play a pivotal role in conveying information, setting the mood, and creating an immersive user experience. Flutter supports various image formats and provides mechanisms for both local and network-based image integration.

Principles:

1. Organizing Image Assets: Structuring image assets within the 'assets' folder and updating the pubspec.yaml file facilitates proper asset management. This step ensures clarity and ease of access for images within the Flutter project.
2. Loading Images: Utilizing the Image.asset widget for local images and Image.network for network images simplifies the integration process. This allows developers to incorporate visuals seamlessly into their application interfaces.

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

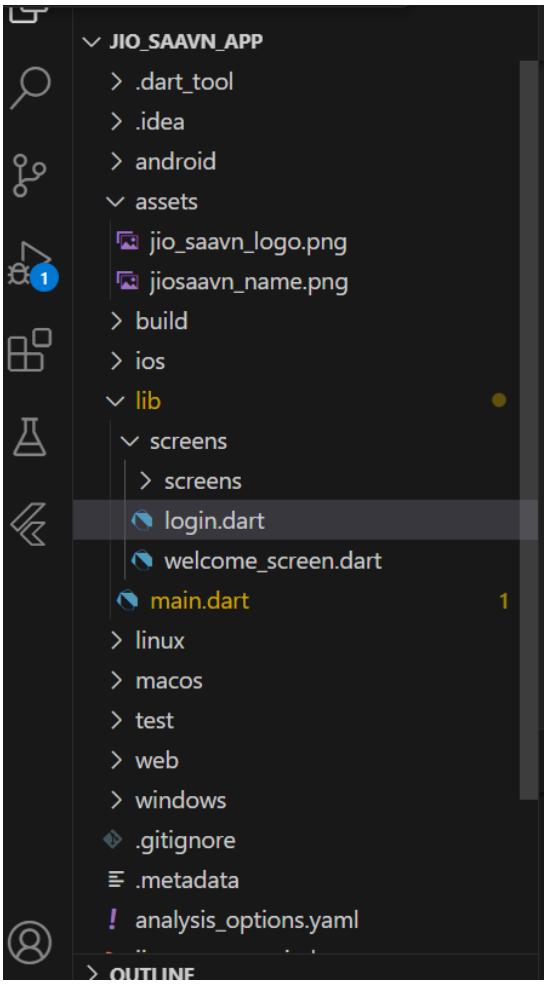
Fonts

Custom fonts contribute to brand identity and enhance the overall aesthetics of an application. Flutter enables the integration of custom fonts, allowing developers to align the typography with the app's design principles.

Principles:

1. Adding Font Files: Placing font files within a 'fonts' folder and updating the pubspec.yaml file establishes a clear structure for font assets. This practice simplifies the inclusion of custom fonts in Flutter projects.
2. Loading and Applying Fonts: Integrating custom fonts involves specifying the font family and style using the TextStyle widget. This ensures consistent and visually appealing text elements throughout the application.

File structure:



Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

Code:

//main.dart

```dart
import 'package:flutter/material.dart';
import 'screens/login.dart';
import 'screens/welcome_screen.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'JioSaavn App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: const WelcomePage(),
    );
  }
}
```

//login.dart

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: LoginScreen(),
    );
  }
}

class LoginScreen extends StatelessWidget {
  @override
```

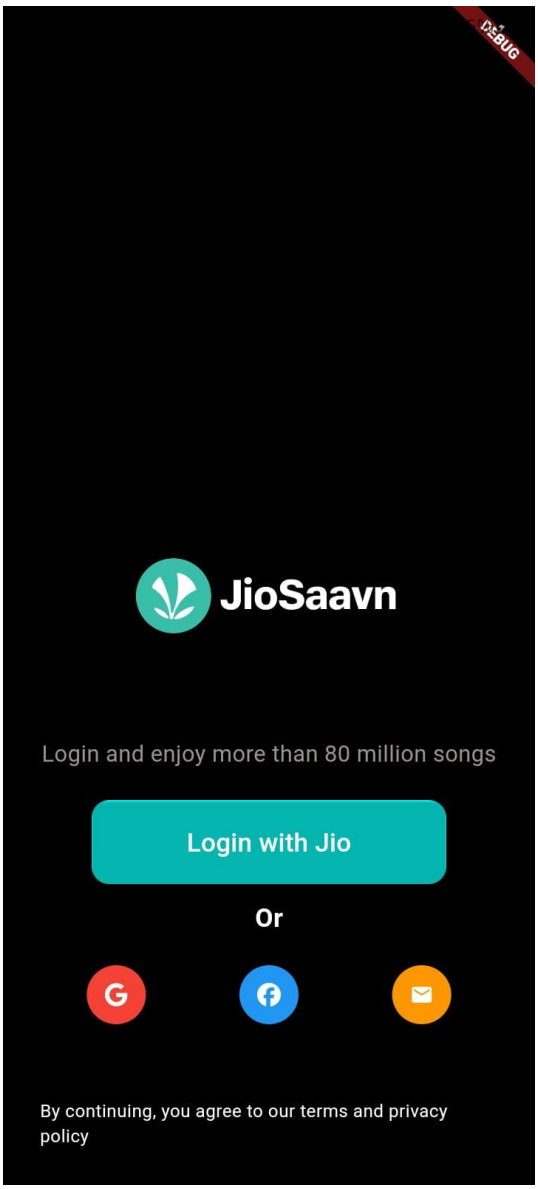
```
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      body: Column(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          Padding(
            padding: const EdgeInsets.all(25.0),
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                // Jio Saavn Logo
                Image.asset(
                  'assets/jiosaavn_name.png', // Path to Jio Saavn logo
                  height: 150,
                  width: 200,
                ),
                SizedBox(height: 20),

                Text(
                  'Login and enjoy more than 80 million songs',
                  style: TextStyle(
                    fontSize: 14,
                    fontWeight: FontWeight.normal,
                    color: Color.fromARGB(198, 202, 189, 189),
                  ),
                ),
                SizedBox(height: 20),

                // Login with Jio Button
                Container(
                  width: double.infinity,
                  margin: EdgeInsets.symmetric(horizontal: 35),
                  decoration: BoxDecoration(
                    color: Color.fromRGBO(0, 255, 240, 1),
                    borderRadius: BorderRadius.circular(12),
                  ),
                  child: ElevatedButton(
                    onPressed: () {
                      // Handle Jio login
                    },
                    style: ElevatedButton.styleFrom(
                      primary: Colors
                          .transparent, // Transparent color for elevated
button
                      shape: RoundedRectangleBorder(
                        borderRadius: BorderRadius.circular(8),
                      ),
```

```
                    ),
                    child: Padding(
                      padding: const EdgeInsets.all(16.0),
                      child: Text(
                        "Login with Jio",
                        style: TextStyle(fontSize: 16, color: Colors.white),
                      ),
                    ),
                  ),
                ),
                SizedBox(height: 10),

                Text(
                  'Or',
                  style: TextStyle(
                    fontSize: 16,
                    fontWeight: FontWeight.bold,
                    color: Colors.white,
                  ),
                ),
                SizedBox(height: 20),

                // Social Logins
                Row(
                  mainAxisAlignment: MainAxisAlignment.spaceAround,
                  children: [
                    _buildSocialLoginButton(
                        FontAwesomeIcons.google, "", Colors.red, () {
                      // Handle Google login
                    }),
                    _buildSocialLoginButton(
                        FontAwesomeIcons.facebook, "", Colors.blue, () {
                      // Handle Facebook login
                    }),
                    _buildSocialLoginButton(Icons.mail, "", Colors.orange, ()
{
                      // Handle Mail login
                    }),
                  ],
                ),
                SizedBox(height: 20),

                Text(
                  'By continuing, you agree to our terms and privacy policy',
                  style: TextStyle(
                    fontSize: 11,
                    fontWeight: FontWeight.normal,
                    color: Colors.white,
```

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

Widget _buildSocialLoginButton(
    IconData icon, String text, Color color, VoidCallback onPressed) {
  return InkWell(
    onTap: onPressed,
    child: Column(
      children: [
        CircleAvatar(
          radius: 20, // Reduced size
          backgroundColor: color,
          child: Icon(icon, size: 16, color: Colors.white),
        ),
        SizedBox(height: 8),
        Text(
          text,
          style: TextStyle(color: Colors.white),
        ),
      ],
    ),
  );
}
}
```

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

Output:



Conclusion:

In conclusion, the integration of icons, images, and fonts in Flutter applications is foundational to creating visually stunning and engaging user interfaces. By adhering to best practices, developers can uphold design consistency, enhance user experience, and bring forth applications that resonate aesthetically with their target audience.

MAD & PWA Lab

Journal

Experiment No.	04
Experiment Title.	To create an interactive Form using form widget
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

MAD PWA Lab Exp 4

Aim: To create an interactive Form using form widget

Theory:

Forms play a vital role in collecting and processing user input in mobile applications. Flutter provides a powerful Form widget that simplifies the process of creating interactive and responsive forms. This theory covers the fundamental concepts and practices involved in creating an interactive form using the Form widget.

Key Concepts:

Form Widget:
The Form widget serves as a container for form elements, helping to organize and manage user input.
It enables the grouping of various form fields and facilitates form-level operations like validation and submission.

TextFormFields:
Text input is a common element in forms, and Flutter provides the TextFormField widget for this purpose.
TextFormField allows developers to collect textual input from users and supports customization through attributes like decoration and validator.

Form Validation:
The Form widget includes a powerful validation mechanism using the validator property in form fields.
Developers can define custom validation functions to ensure that user input meets specific criteria before submission.

Form Submission:
Forms typically include a mechanism for submitting user input. Flutter provides the onSaved property to handle the form data once it passes validation.
Developers can implement custom logic to process and store the validated form data.

Form Key:
Managing the form's state is crucial for interactivity. Flutter uses a GlobalKey<FormState> to interact with the form's state.

The key enables actions such as resetting the form, validating all form fields, and accessing the form's current state.

Best Practices:
Organizing Form Elements:
Structure the form using a Column or ListView for better organization and readability.
Group related form fields together to enhance user experience.

Error Handling:
Leverage the validator property to provide meaningful error messages to users when form input is invalid.
Display errors in a user-friendly manner, enhancing the overall user experience.

Responsive Design:
Ensure that the form is responsive to different screen sizes and orientations.
Use Flutter's layout widgets to create a consistent and visually appealing form across devices.

Accessibility:
Implement accessibility features to ensure users with disabilities can interact with and understand the form.
Use proper labels, hints, and error messages to enhance accessibility.

Code & Implementation:

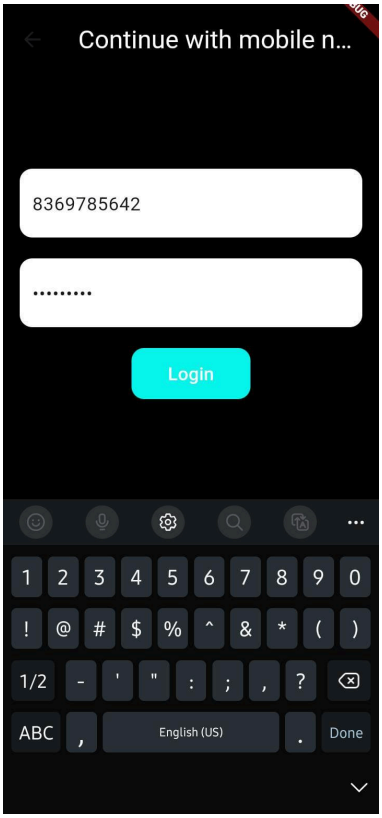
//mobilenumber.dart

```dart
import 'package:flutter/material.dart';
import 'package:jio_saavn_app/screens/home.dart';

class MobileNumberPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(
          'Continue with mobile number',
          style: TextStyle(color: Colors.white),
        ),
```

```
      backgroundColor: Colors.black,
    ),
    backgroundColor: Colors.black,
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          TextField(
            keyboardType: TextInputType.phone,
            decoration: InputDecoration(
              hintText: 'Enter mobile number',
              fillColor: Colors.white,
              filled: true,
              border: OutlineInputBorder(
                borderRadius: BorderRadius.circular(12.0),
                borderSide: BorderSide.none,
              ),
            ),
          ),
          SizedBox(height: 20),

          // Password TextField
          TextField(
            obscureText: true, // Hide the entered text for passwords
            decoration: InputDecoration(
              hintText: 'Enter password',
              fillColor: Colors.white,
              filled: true,
              border: OutlineInputBorder(
                borderRadius: BorderRadius.circular(12.0),
                borderSide: BorderSide.none,
              ),
            ),
          ),
          SizedBox(height: 20),

          ElevatedButton(
            onPressed: () {
              Navigator.push(
```

```
                  context,
                  MaterialPageRoute(builder: (context) => HomeScreen()),
                );
                // Handle mobile number login
                // You can implement your logic to verify the mobile
number and navigate to the next screen
              },
              style: ElevatedButton.styleFrom(
                primary: Color.fromRGBO(0, 255, 240, 1), // Aqua color
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(12.0),
                ),
              ),
              child: Padding(
                padding: const EdgeInsets.all(12.0),
                child: Text(
                  'Login',
                  style: TextStyle(fontSize: 16, color: Colors.white),
                ),
              ),
            ),
],
        ),
      ),
    );
  }
}
```

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'



Conclusion: Creating interactive forms in Flutter involves leveraging the Form widget, managing form state, and incorporating best practices for a seamless user experience. By following these principles, developers can design robust and user-friendly forms in their Flutter applications.

MAD & PWA Lab

Journal

Experiment No.	05
Experiment Title.	To apply navigation, routing and gestures in Flutter App
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

MAD PWA Lab Exp 5

Aim: To apply navigation, routing and gestures in Flutter app.

Theory:

In the realm of Flutter app development, the effective application of navigation, routing, and gestures is pivotal for creating an intuitive and engaging user experience. This theoretical exploration delves into the foundational concepts and principles guiding the implementation of navigation, routing, and gestures in Flutter applications.

Navigation and Routing

Navigation is the process of moving between different screens or sections within an app. Flutter employs a structured routing system to facilitate seamless transitions between these screens. This system is driven by the following principles:

Navigator.push()

The Navigator.push() method is employed to navigate from one screen to another by adding a new page to the navigation stack. This method is integral to the dynamic flow of a Flutter application.

Principles:

1. Screen Transition:

Invoking Navigator.push() triggers a transition to a new screen, pushing it onto the navigation stack.

This method is typically used when moving from one screen to another, such as transitioning from a home screen to a details screen.

2. Named Routes:

Navigator.pushNamed() is a variation of this method used when navigating to a screen defined by a named route.

Named routes provide a clear and maintainable way to organize and navigate between different screens in the app.

3. Arguments:

The method allows the passing of arguments to the destination screen, enabling dynamic content rendering.

Arguments facilitate the customization of the new screen based on the context of the navigation.

Navigator.pop()

Conversely, the Navigator.pop() method is employed to remove the current screen from the navigation stack, returning to the previous screen. It plays a vital role in controlling the flow of the application.

Principles:

1. Screen Removal:

When Navigator.pop() is called, the current screen is popped off the navigation stack, reverting to the previous screen.

This method is typically used in scenarios where a user completes a task on one screen and returns to the previous screen.

2. Data Passing:

Information can be passed back to the previous screen using the Navigator.pop() method.

This allows for a seamless exchange of data between screens, enhancing the overall user experience.

3. Return Values:

Developers can retrieve values returned from the popped screen, enabling dynamic updates or actions based on the user's interactions.

Code:

```
//home.dart

import 'package:flutter/material.dart';

import 'package:jio_saavn_app/screens/library.dart';

import 'package:jio_saavn_app/screens/search.dart';
```

```
import 'navigation_bar.dart'; // Import the custom navigation bar

class HomeScreen extends StatefulWidget {
  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  int _selectedIndex = 0;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(12.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Center(
                child: Align(
                  alignment: Alignment.topLeft,
                  child: Text(
                    'Music',
                    style: TextStyle(
                      fontSize: 30,
                      fontWeight: FontWeight.bold,
                      color: Color.fromARGB(255, 255, 255, 255),
                    ),
                  ),
```

```
  ),
),
SizedBox(height: 10),
SizedBox(height: 20),
Text(
  'Recommended Songs for you:',
  style: TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Color.fromARGB(255, 255, 255, 255),
  ),
),
SizedBox(height: 25),
_buildImageRow([
  'Alone',
  'Kesariya',
  'We own it',
], [
  'assets/song1.png',
  'assets/song2.png',
  'assets/song3.png',
]),
SizedBox(height: 20),
Text(
  'Top artists for you:',
  style: TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Color.fromARGB(255, 255, 255, 255),
  ),
),
```

```
SizedBox(height: 25),
_buildImageRow([
  'Eminem',
  'Arjit Singh',
  'Selena Gomez',
], [
  'assets/eminem.png',
  'assets/arjit.png',
  'assets/selena.png',
]),
SizedBox(height: 20),
Text(
  'New Releases:',
  style: TextStyle(
    fontSize: 20,
    fontWeight: FontWeight.bold,
    color: Color.fromARGB(255, 255, 255, 255),
  ),
),
SizedBox(height: 25),
_buildImageRow([
  'Alone',
  'Kesariya',
  'We own it',
  'We own it',
], [
  'assets/song1.png',
  'assets/song2.png',
  'assets/song3.png',
  'assets/song3.png',
]),
```

```
        SizedBox(height: 20),
        Text(
          'Jio Saavn picks:',
          style: TextStyle(
            fontSize: 20,
            fontWeight: FontWeight.bold,
            color: Color.fromARGB(255, 255, 255, 255),
          ),
        ),
        SizedBox(height: 25),
        _buildImageRow([
          'Alone',
          'Kesariya',
          'We own it',
        ], [
          'assets/song1.png',
          'assets/song2.png',
          'assets/song3.png',
        ]),
      ],
    ),
  ),
),
bottomNavigationBar: CustomBottomNavigationBar(
  selectedIndex: _selectedIndex,
  onItemTapped: (index) {
    setState(() {
      _selectedIndex = index;
    });
    if (_selectedIndex == 1) {
      // Navigate to SearchScreen when the Search tab is tapped
```

```
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => SearchScreen()),
          );
        } else if (_selectedIndex == 2) {
          // Navigate to MyLibrary when the Library tab is tapped
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => MyLibrary()),
          );
        }
      },
    ),
  );
}

Color _getColor(int index) {
  return _selectedIndex == index
      ? Colors.white
      : const Color.fromARGB(255, 128, 128, 128);
}

Widget _buildSuggestedItem(
    BuildContext context, String name, String imagePath) {
  return GestureDetector(
    onTap: () {
      // Navigate to SongDetailsScreen when the image is tapped
      //  Navigator.push(
      //    context,
      //    MaterialPageRoute(
      //      builder: (context) => SongDetailsScreen(songTitle: name, songImagePath: imagePath),
```

```
      //    ),
      //  );
    },
    child: Container(
      width: 120,
      margin: EdgeInsets.only(right: 5),
      child: Column(
        children: [
          Image.asset(
            imagePath,
            height: 100,
            width: 100,
            fit: BoxFit.cover,
          ),
          SizedBox(height: 5),
          Text(
            name,
            style: TextStyle(color: Colors.white),
          ),
        ],
      ),
    ),
  );
}

 Widget _buildImageRow(List<String> names, List<String> imagePaths) {
  return SizedBox(
    height: 150,
    child: ListView.builder(
      scrollDirection: Axis.horizontal,
      itemCount: names.length,
```

```
      itemBuilder: (BuildContext context, int index) {
        return _buildSuggestedItem(
          context,
          names[index],
          imagePaths[index],
        );
      },
    ),
  );
 }
}

//search.dart
import 'package:flutter/material.dart';
import 'package:jio_saavn_app/screens/library.dart';
import 'home.dart';
import 'navigation_bar.dart'; // Import your custom navigation bar

class SearchScreen extends StatefulWidget {
 @override
 _SearchScreenState createState() => _SearchScreenState();
}

class _SearchScreenState extends State<SearchScreen> {
 int _selectedIndex = 1; // Set the default index to Search

 @override
 Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black, // Set background color here
```

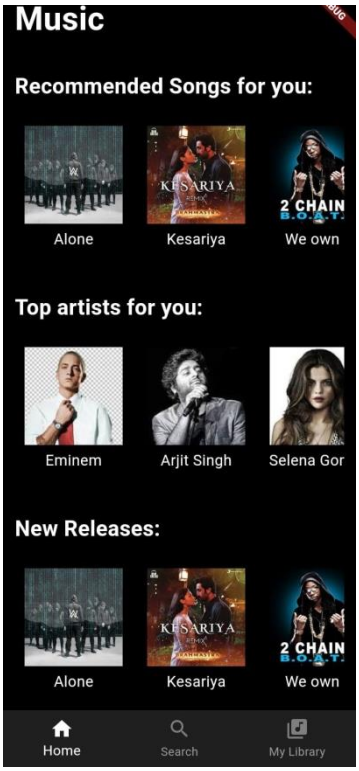
```
body: SingleChildScrollView(
 child: Padding(
  padding: const EdgeInsets.all(16.0),
  child: Column(
   crossAxisAlignment: CrossAxisAlignment.start,
   children: [
    // Add your search bar here
    _buildSearchBar(),

    // Add the top trending songs title
    Text(
     'Trending',
     style: TextStyle(
      fontSize: 24,
      fontWeight: FontWeight.bold,
      color: Color.fromARGB(255, 255, 255, 255),
     ),
    ),

    // Add the three images with their song names
    _buildSuggestedItem('Song 4', 'assets/song4.png'),
    _buildSuggestedItem('Song 5', 'assets/song5.png'),
    _buildSuggestedItem('Song 6', 'assets/song6.png'),
   ],
  ),
 ),
),
bottomNavigationBar: CustomBottomNavigationBar(
 selectedIndex: _selectedIndex,
 onItemTapped: (index) {
  setState(() {
```

```
        _selectedIndex = index;
       });
       if (_selectedIndex == 0) {
        // Navigate to HomeScreen when the Home tab is tapped
        Navigator.push(
         context,
         MaterialPageRoute(builder: (context) => HomeScreen()),
        );
       } else if (_selectedIndex == 2) {
        // Navigate to MyLibrary when the Library tab is tapped
        Navigator.push(
         context,
         MaterialPageRoute(builder: (context) => MyLibrary()),
        );
       }
      },
      currentIndex: _selectedIndex,
    ),
  );
}

Color _getColor(int index) {
  return _selectedIndex == index
      ? Colors.white
      : const Color.fromARGB(255, 128, 128, 128);
}

Widget _buildSearchBar() {
  return Container(
    margin: EdgeInsets.symmetric(vertical: 16),
    padding: EdgeInsets.symmetric(horizontal: 16),
```

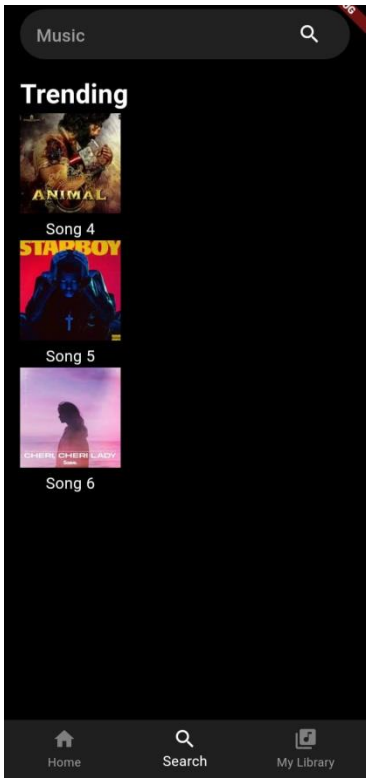
```
    decoration: BoxDecoration(
      color: Colors.grey[900],
      borderRadius: BorderRadius.circular(30),
    ),
    child: TextField(
      style: TextStyle(color: Colors.white),
      decoration: InputDecoration(
        hintText: 'Music',
        hintStyle: TextStyle(color: Colors.grey),
        suffixIcon: Icon(Icons.search, color: Colors.white),
        border: InputBorder.none,
      ),
    ),
  );
}

Widget _buildSuggestedItem(String name, String imagePath) {
  return Column(
    children: [
      Image.asset(
        imagePath,
        height: 100, // Set the desired height
        width: 100, // Set the desired width
        fit: BoxFit.cover, // Adjust the image size
      ),
      SizedBox(height: 5),
      Text(
        name,
        style: TextStyle(color: Colors.white),
      ),
    ],
```

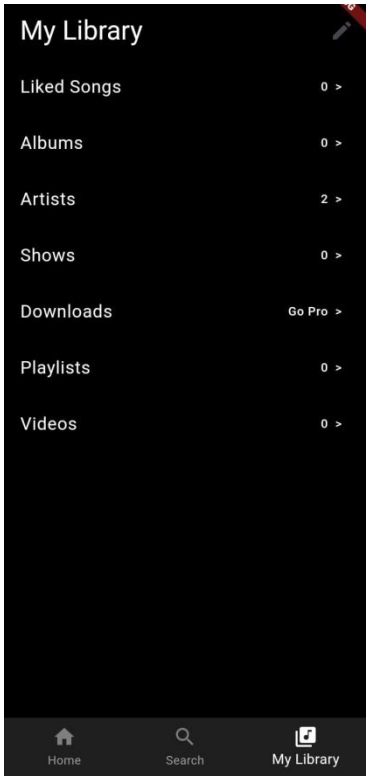
```
    );
  }
}
```

Output:


Music
Recommended Songs for you:
Alone
Kesariya
We own
Top artists for you:
Eminem
Arjit Singh
Selena Gor
New Releases:
Alone
Kesariya
We own
Home
Search
My Library

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'





Conclusion:

In conclusion, Navigator.push() and Navigator.pop() are foundational methods in Flutter navigation, facilitating the seamless movement between screens. Navigator.push() initiates

the addition of a new screen to the stack, allowing for dynamic navigation, while Navigator.pop() removes the current screen, providing a mechanism for controlled navigation and data exchange between screens.

MAD & PWA Lab

Journal

Experiment No.	06
Experiment Title.	To Connect Flutter UI with fireBase database
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	15

Name: Atharva Chavan Div: D15A Roll no: 10 Batch ‘A’

MAD PWA Lab Exp 6

Aim: To Set Up Firebase with Flutter for iOS and Android Apps

Theory:

Firebase is a powerful platform provided by Google that offers a variety of services for mobile and web applications, including real-time databases, authentication, cloud functions, and more. When combined with Flutter, a popular open-source UI software development toolkit, developers can create cross-platform apps for iOS and Android efficiently. This guide will walk you through the process of setting up Firebase with Flutter for both iOS and Android platforms.

Prerequisites

Before you begin, make sure you have the following prerequisites:

Flutter Installed: Ensure that you have Flutter and Dart installed on your development machine.

Firebase Account: Create a Firebase account if you don't have one already. Visit the Firebase Console to set up a new project.

Flutter IDE: Use an integrated development environment (IDE) like Visual Studio Code or Android Studio with Flutter and Dart plugins installed.

Step 1: Create a Firebase Project

1.1. Go to the Firebase Console, click on "Add Project," and follow the prompts to create a new project.
1.2. Once the project is created, click on "Add App" to add both iOS and Android apps to your project.

Step 2: Configure iOS App

2.1. For iOS, click on the iOS icon in the Firebase Console and follow the setup instructions. Download the GoogleService-Info.plist file and add it to your Flutter project's ios/Runner directory.
2.2. Update your ios/Podfile to include the necessary Firebase dependencies. Run pod install in the ios directory.

Step 3: Configure Android App

3.1. For Android, click on the Android icon in the Firebase Console and follow the setup instructions. Download the google-services.json file and add it to your Flutter project's android/app directory.

3.2. Update your android/build.gradle and android/app/build.gradle files with the necessary dependencies.

Step 4: Add FlutterFire Plugins

4.1. Open your pubspec.yaml file and add the necessary FlutterFire plugins for the Firebase services you want to use, such as authentication, Firestore, or Cloud Messaging.
4.2. Run flutter pub get in your terminal to fetch the dependencies.

Step 5: Initialize Firebase in Your Flutter App

5.1. Import the Firebase packages in your Dart code and initialize Firebase in the main.dart file.

Step 6: Test Firebase Integration

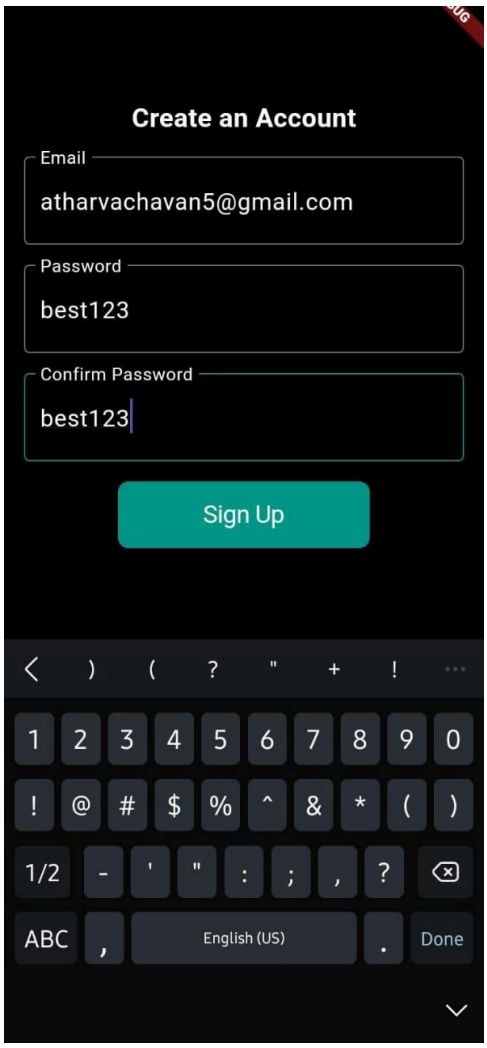
6.1. Write a simple test to ensure Firebase is correctly integrated. For example, try initializing Firestore or authenticate a user.

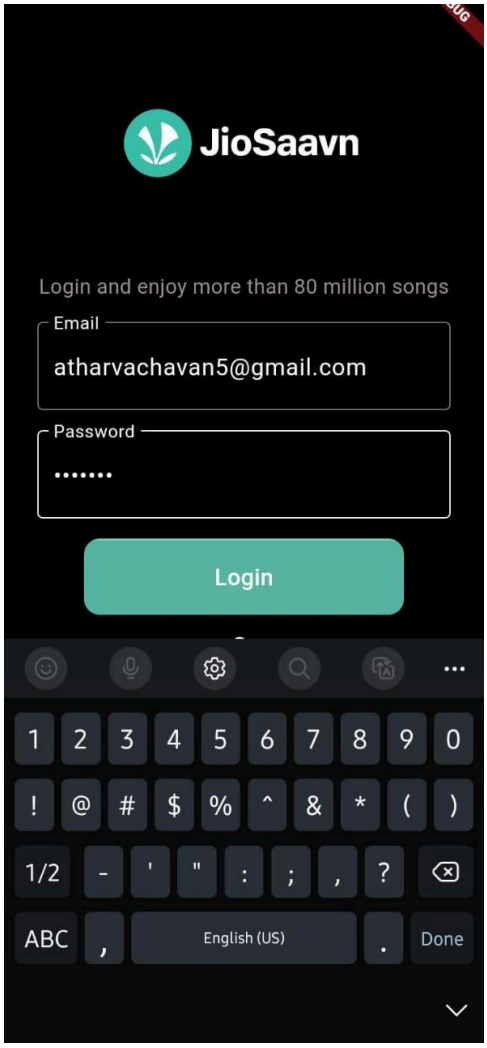
Code & Implementation:

```
//main.dart
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:jio_saavn_auth/firebase_options.dart';
import 'package:jio_saavn_auth/screens/library_screen.dart';
import 'package:jio_saavn_auth/screens/welcome_screen.dart';
import 'package:provider/provider.dart';

Future<void> main() async {
   WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
   options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(
   ChangeNotifierProvider(
     create: (context) => LikedSongsModel(),
     child: MyApp(),
   ),
  );
```

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'

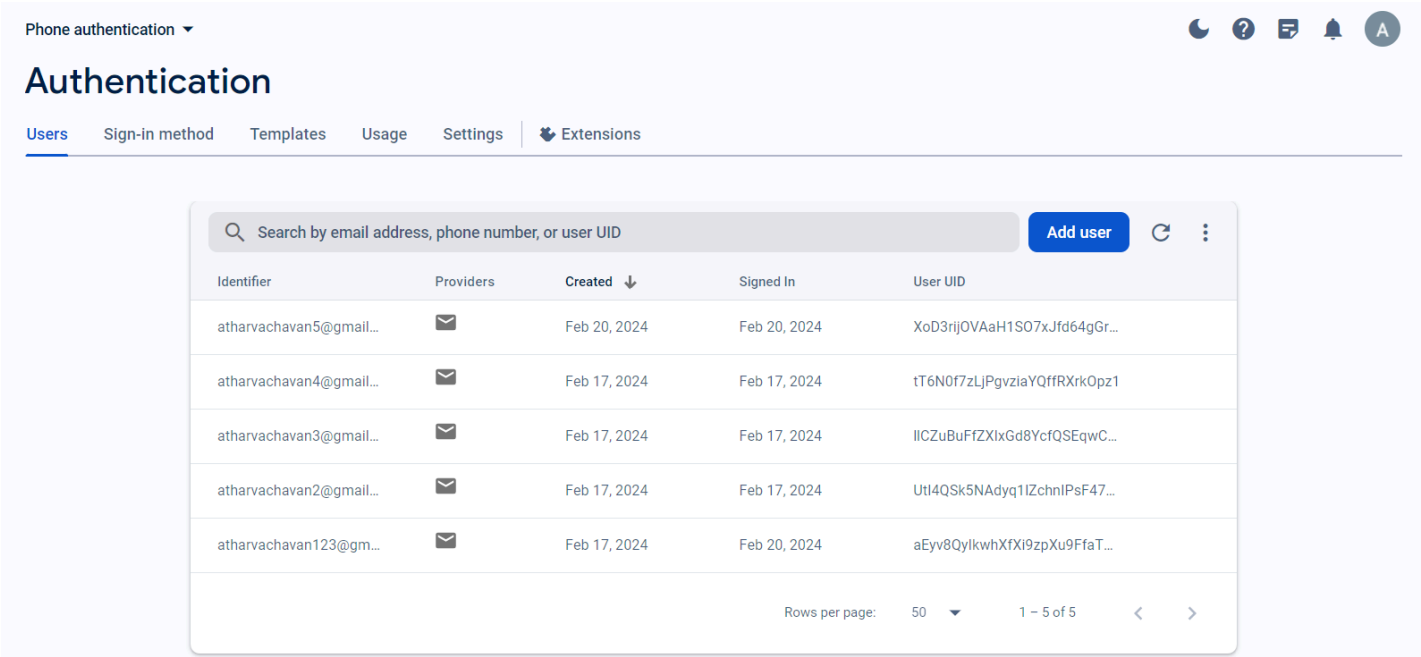
}





Sign Up Screen **Login screen**

Name: Atharva Chavan Div: D15A Roll no: 10 Batch 'A'



Conclusion:

In conclusion, integrating Firebase with Flutter for iOS and Android apps provides a robust foundation for building feature-rich, cross-platform applications. By following the steps outlined in this guide, you have established a seamless connection between your Flutter project and Firebase, unlocking a plethora of services such as authentication, real-time databases, and cloud functions. This integration not only enhances the app's functionality but also simplifies the management of backend services, making development more efficient.

MAD & PWA Lab

Journal

Experiment No.	07
Experiment Title.	To write meta data of your Ecommerce PWA in a Web app manifest file to enable “add to homescreen feature”.
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4: Understand various PWA frameworks and their requirements
Grade:	15

MAD and PWA Lab

Name: Atharva Chavan **Class: D15A** **Roll no:10**

Experiment - 7

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

Theory:

Regular Web App

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

Progressive Web App

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

Difference between PWAs vs. Regular Web Apps:

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

1. Native Experience

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

2. Ease of Access

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

3. Faster Services

PWAs can cache the data and serve the user with text stylesheets, images, and other web content even before the page loads completely. This lowers the waiting time for the end-users and helps the

brands improve the user engagement and retention rate, which eventually adds value to their business.

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

5. Updated Real-Time Data Access
Another plus point of PWAs is that these apps get updated on their own. They do not demand the 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.

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

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

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

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

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

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

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

Code:
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <!-- CSS LINK -->
    <link rel="stylesheet" href="style.css">
    <!-- Fontawesome -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css"
integrity="sha512-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZq
Xgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />
    <title>Cok Sayfali Web Sitesi | Nizami Sevindi</title>
</head>
    <!-- Google tag (gtag.js) -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-PX8CKQHBTF"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-PX8CKQHBTF');
</script>
<body>
    <!------------------------------------------- HEADER SECTION -->
    <header class="header" >
        <a href="#" class="logo">
            <img src="./image/pngegg.png" alt="logo">
        </a>
        <nav class="navbar">
            <a href="./index.html" class="active">home</a>
            <a href="./about.html">about</a>
            <a href="./menu.html">menu</a>
            <a href="./products.html">products</a>
            <a href="./review.html">review</a>
            <a href="./contact.html">contact</a>
            <a href="./blog.html">blog</a>
        </nav>
        <div class="buttons">
            <button id="search-btn">
                <i class="fas fa-search"></i>
            </button>
            <button id="cart-btn">
                <i class="fas fa-shopping-cart"></i>
            </button>
            <button id="menu-btn">
                <i class="fas fa-bars"></i>
            </button>
        </div>
        <div class="search-form">
            <input type="text" class="search-input" id="search-box" placeholder="Search">
            <i class="fas fa-search"></i>
        </div>
        <div class="cart-items-container">
            <div class="cart-item">
                <i class="fas fa-times"></i>
                <img src="./image/pngwing.com (1).png" alt="menu">
```

```
            <div class="content">
                <h3>cart item 01</h3>
                <div class="price">$15.99 </div>
            </div>
        </div>
        <div class="cart-item">
            <i class="fas fa-times"></i>
            <img src="./image/pngwing.com (2).png" alt="menu">
            <div class="content">
                <h3>cart item 02</h3>
                <div class="price">$16.99 </div>
            </div>
        </div>
        <div class="cart-item">
            <i class="fas fa-times"></i>
            <img src="./image/pngwing.com (3).png" alt="menu">
            <div class="content">
                <h3>cart item 03</h3>
                <div class="price">$13.99 </div>
            </div>
        </div>
        <div class="cart-item">
            <i class="fas fa-times"></i>
            <img src="./image/pngwing.com (4).png" alt="menu">
            <div class="content">
                <h3>cart item 04</h3>
                <div class="price">$12.99 </div>
            </div>
        </div>
        <a href="#" class="btn">check out </a>
    </div>
</header>
<!----------------------------------------------HEADER SECTION -->

<!----------------------------------------------HOME SECTION -->
<section class="home" id="home">
    <div class="content">
    <h3>Fast Food Delivery</h3>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt perferendis obcaecati iste
voluptatum, quaerat nihil magnam numquam sint? </p>
    <a href="#" class="btn">order now</a>
    </div>
</section>
<!----------------------------------------------HOME SECTION -->

<!----------------------------------------------MENU SECTION -->
<section class="menu" id="menu">
    <h1 class="heading">our <span>menu</span></h1>
    <div class="box-container">
        <div class="box">

            <div class="box-head">
```

```
            <img src="./image/pngwing.com.png" alt="">
            <span class="menu-category">Pizza</span>
            <h3>6 Mini Pizzas</h3>
            <div class="price">$104.99 <span>$119.99</span></div>
        </div>
        <div class="box-bottom">
            <a href="#" class="btn">add to cart</a>
        </div>
    </div>
    <div class="box">

        <div class="box-head">
            <img src="./image/pngwing.com (2).png" alt="">
            <span class="menu-category">Burger</span>
            <h3>5 Mini Burgers</h3>
            <div class="price">$99.99 <span>$105.99</span></div>
        </div>
        <div class="box-bottom">
            <a href="#" class="btn">add to cart</a>
        </div>
    </div>
    <div class="box">

        <div class="box-head">
            <img src="./image/pngwing.com (1).png" alt="">
            <span class="menu-category">Pizza</span>
            <h3>2 Mixed Pizzas</h3>
            <div class="price">$49.99 <span>$59.99</span></div>
        </div>
        <div class="box-bottom">
            <a href="#" class="btn">add to cart</a>
        </div>
    </div>
    <div class="box">

        <div class="box-head">
            <img src="./image/pngwing.com (3).png" alt="">
            <span class="menu-category">Burger</span>
            <h3>3 Meatball Burgers</h3>
            <div class="price">$79.99 <span>$99.99</span></div>
        </div>
        <div class="box-bottom">
            <a href="#" class="btn">add to cart</a>
        </div>
    </div>
  </div>
</section>
<!-----------------------------------------MENU SECTION -->

<!-----------------------------------------PRODUCTS SECTION -->
<section class="products" id="products">
  <h1 class="heading">our <span>products</span> </h1>
```

```
<div class="box-container">
    <div class="box">
        <div class="box-head">
            <span class="title">mini burger</span>
            <a href="#" class="name">Bacon Burger</a>
        </div>
        <div class="image">
            <img src="./image/pngwing.com (6).png" alt="">
        </div>
        <div class="box-bottom">
            <div class="info">
                <b class="price">$6.00</b>
                <span class="amount">110gr / 300 Cal</span>
            </div>
            <div class="product-btn">
                <a href="#">
                    <i class="fas fa-plus"></i>
                </a>
            </div>
        </div>
    </div>
    <div class="box">
        <div class="box-head">
            <span class="title">cheese burger</span>
            <a href="#" class="name">cheese Burger</a>
        </div>
        <div class="image">
            <img src="./image/pngwing.com (4).png" alt="">
        </div>
        <div class="box-bottom">
            <div class="info">
                <b class="price">$12.00</b>
                <span class="amount">140gr / 2500 Cal</span>
            </div>
            <div class="product-btn">
                <a href="#">
                    <i class="fas fa-plus"></i>
                </a>
            </div>
        </div>
    </div>
    <div class="box">
        <div class="box-head">
            <span class="title">Double burger</span>
            <a href="#" class="name">Double Burger</a>
        </div>
        <div class="image">
            <img src="./image/pngwing.com (5).png" alt="">
        </div>
        <div class="box-bottom">
            <div class="info">
                <b class="price">$24.00</b>
```

```
                <span class="amount">440gr / 600 Cal</span>
            </div>
            <div class="product-btn">
                <a href="#">
                    <i class="fas fa-plus"></i>
                </a>
            </div>
        </div>
    </div>
</section>
<!----------------------------------------------PRODUCTS SECTION -->

<!----------------------------------------------ABOUT US SECTION -->
<section class="about" id="about">
    <h1 class="heading">about <span>us</span> </h1>

    <div class="row">
        <div class="image">
            <img src="./image/pngwing.com (7).png" alt="">
        </div>
        <div class="content">
            <h3>What is the secret receipe of our burgers</h3>
            <div class="paragraph">
                <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quia officia id et, corrupti
assumenda.</p>
                <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quia officia id et, corrupti
assumenda.</p>
                <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. Quia officia id et, corrupti
assumenda.</p>
            </div>
                <a href="#" class="btn">Learn More</a>
        </div>
    </div>
</section>
<!----------------------------------------------ABOUT US SECTION -->

<!----------------------------------------------REVIEW SECTION -->
<section class="review" id="review">
    <h1 class="heading">customer's <span>review</span> </h1>
    <div class="box-container">
        <div class="box">
            <img src="./image/pngwing.com (8).png" alt="quote">
            <p> Dicta totam suscipit vero praesentium excepturi facilis, fuga at architecto dolor tempora
molestias quam dignissimos sit. Molestiae temporibus ratione quas placeat possimus!</p>
            <img src="./image/pngwing.com (9).png" alt="customer-avatar" class="user">
            <h3>Patrick Hellinger</h3>
            <div class="stars">
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
```

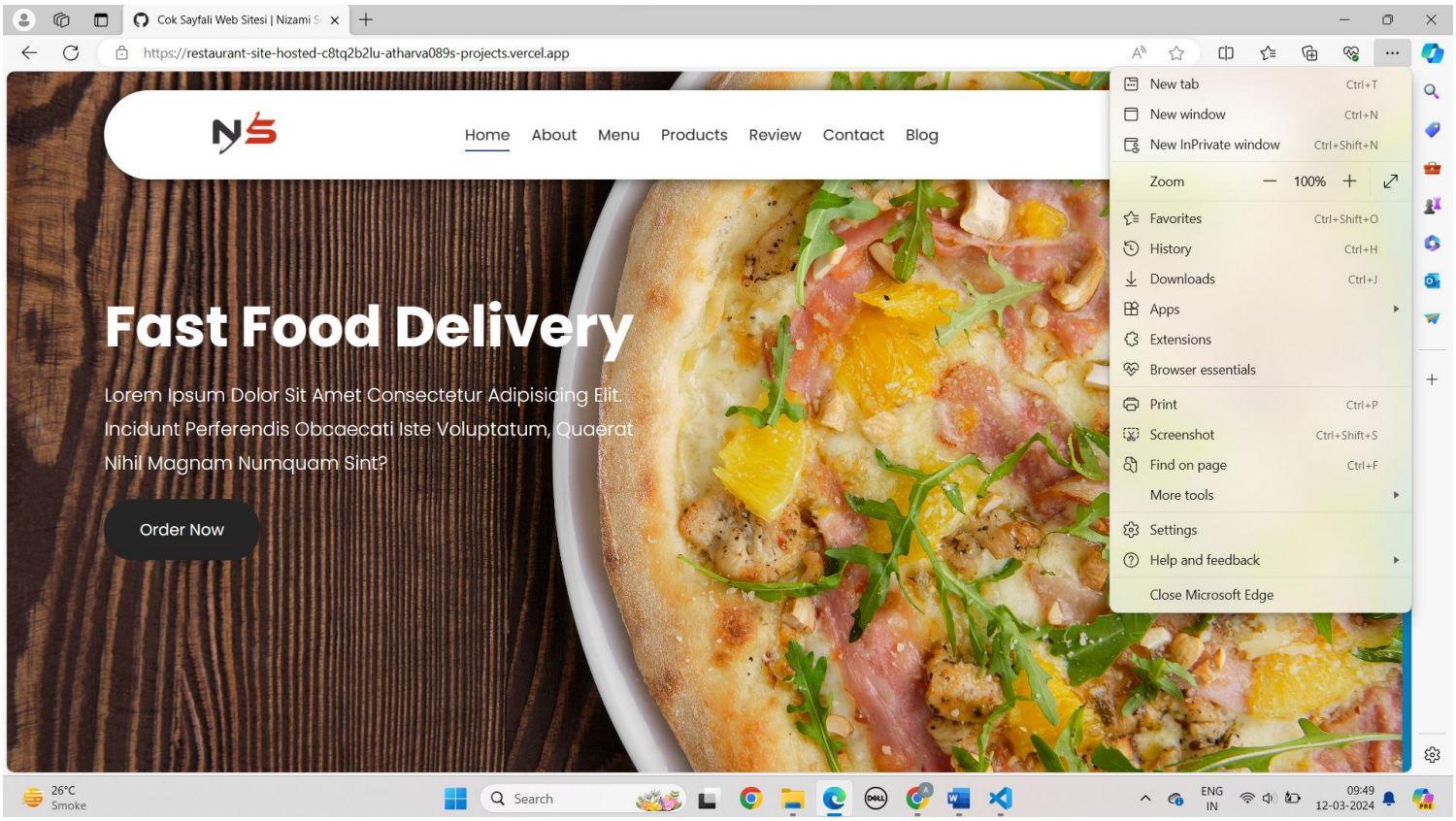
```
                <i class="fas fa-star-half-alt"></i>
            </div>
        </div>
        <div class="box">
            <img src="./image/pngwing.com (8).png" alt="quote">
            <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Fuga at architecto dolor tempora molestias quam dignissimos sit. Molestiae temporibus ratione quas placeat possimus!</p>
            <img src="./image/pngwing.com (10).png" alt="customer-avatar" class="user">
            <h3>Serena Williams</h3>
            <div class="stars">
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star-half-alt"></i>
            </div>
        </div>
        <div class="box">
            <img src="./image/pngwing.com (8).png" alt="quote">
            <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Dicta totam suscipit vero praesentium excepturi facilis, fuga at architecto dolor tempora molestias quam dignissimos possimus!</p>
            <img src="./image/pngwing.com (11).png" alt="customer-avatar" class="user">
            <h3>Helen Marksen</h3>
            <div class="stars">
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star"></i>
                <i class="fas fa-star-half-alt"></i>
            </div>
        </div>
    </div>
</section>
<!----------------------------------------REVIEW SECTION -->
<!----------------------------------------CONTACT SECTION -->
<section class="contact" id="contact">
    <h1 class="heading">contact <span>us</span> </h1>
    <div class="row">
        <iframe class="map" src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d983.3235940970079!2d8.540719273659763!3d47.3713194174677!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x47900a00aa1e1d17%3A0x278f576acdd580f5!2sStorchen%20Z%C3%BCrich%20-%20Lifestyle%20Boutique%20Hotel!5e0!3m2!1sde!2sch!4v1658505945506!5m2!1sde!2sch" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-downgrade"></iframe>
        <form>
            <h3>get in touch</h3>
            <div class="inputBox">
                <i class="fas fa-user"></i>
                <input type="text" placeholder="name">
            </div>
            <div class="inputBox">
```

```
                <i class="fas fa-envelope"></i>
                <input type="email" placeholder="email">
            </div>
            <div class="inputBox">
                <i class="fas fa-phone"></i>
                <input type="number" placeholder="number">
            </div>
            <input type="submit" class="btn" value="contact now">
        </form>
    </div>
</section>
<!----------------------------------------------CONTACT SECTION -->

<!----------------------------------------------BLOG SECTION -->
<section class="blog" id="blog">
    <h1 class="heading">our <span>blog</span> </h1>
    <div class="box-container">
        <div class="box-full">
            <div class="image">
                <img src="./image/delicious-burger-on-wooden-board-2022-03-04-05-58-25-utc.jpg" alt="">
            </div>
            <div class="content">
                <a href="#" class="title">how to make burgers</a>
                <span>by admin / 10st may, 2020</span>
                <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minus eos esse nesciunt
cupiditate expedita.</p>
                <a href="#" class="btn">read more</a>
            </div>
        </div>
        <div class="box-full">
            <div class="image">
                <img src="./image/delicious-tasty-burgers-on-wooden-background-2021-08-26-15-25-13-
utc.jpg" alt="">
            </div>
            <div class="content">
                <a href="#" class="title">how to make burgers</a>
                <span>by admin / 10st may, 2020</span>
                <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minus eos esse nesciunt
cupiditate expedita.</p>
                <a href="#" class="btn">read more</a>
            </div>
        </div>
        <div class="box-full">
            <div class="image">
                <img src="./image/handmade-burger-on-dark-background-delicious-blac-2021-10-21-02-27-
27-utc.jpg" alt="">
            </div>
            <div class="content">
                <a href="#" class="title">how to make burgers</a>
                <span>by admin / 10st may, 2020</span>
                <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minus eos esse nesciunt
cupiditate expedita.</p>
```

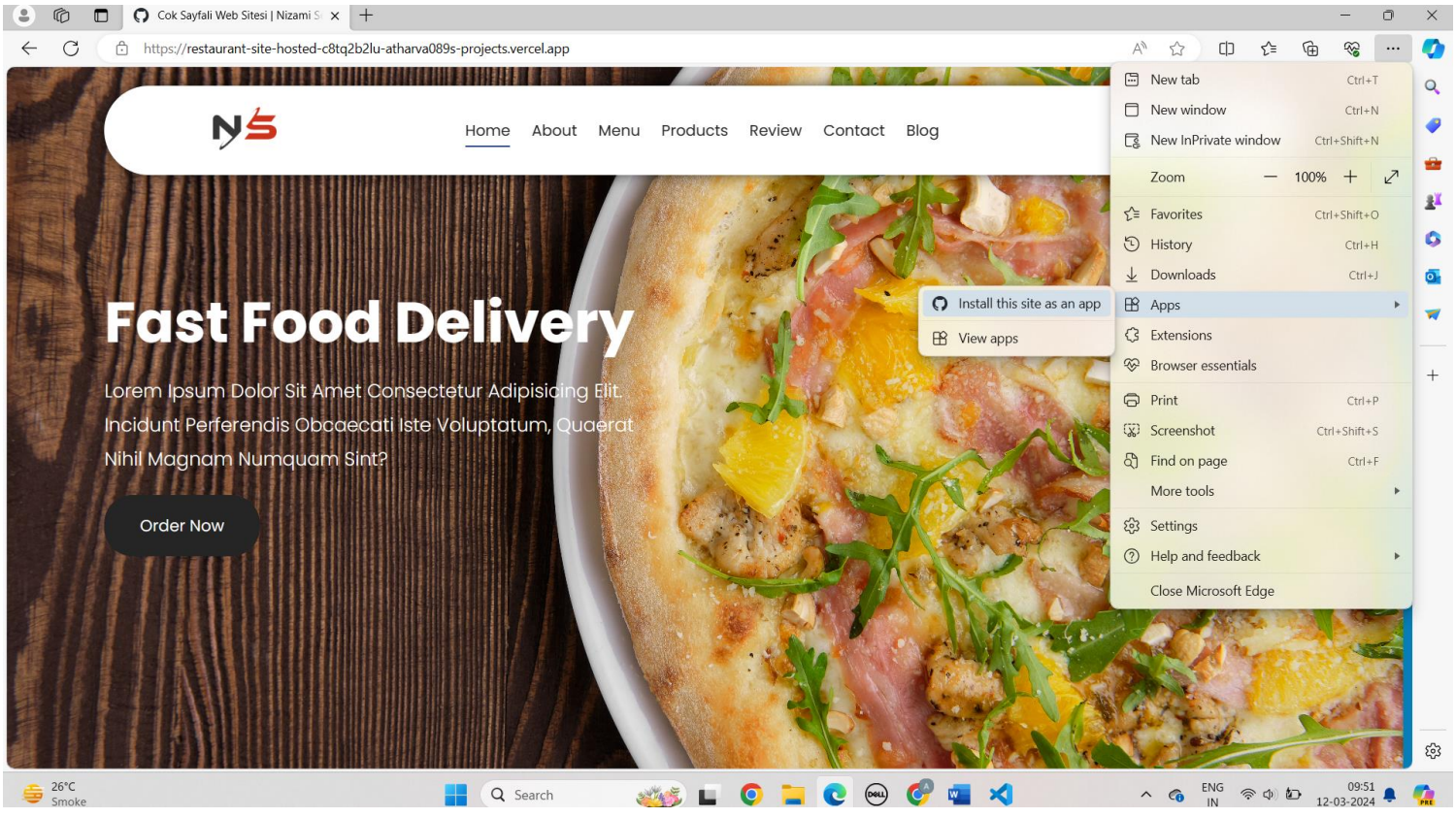
```
                <a href="#" class="btn">read more</a>
            </div>
        </div>
    </div>
</section>
<!----------------------------------------------BLOG SECTION -->

<!----------------------------------------------FOOTER SECTION -->
<section class="footer">
    <div class="search">
        <input type="text" class="search-input" placeholder="Search">
        <button class="btn btn-primary">search</button>
    </div>
    <div class="share">
        <a href="#" class="fab fa-facebook"></a>
        <a href="#" class="fab fa-twitter"></a>
        <a href="#" class="fab fa-instagram"></a>
        <a href="#" class="fab fa-linkedin"></a>
        <a href="#" class="fab fa-pinterest"></a>
    </div>
    <div class="links">
        <a href="#home">home</a>
        <a href="#about">about</a>
        <a href="#menu">menu</a>
        <a href="#products">products</a>
        <a href="#review">review</a>
        <a href="#contact">contact</a>
        <a href="#blog">blog</a>
    </div>
    <div class="credit">
        creaated by <span>Nizami Sevindi</span> | all rights reserved!
    </div>
</section>
<!----------------------------------------------FOOTER SECTION -->

<script src="./script.js"></script>
</body>
</html>
```

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

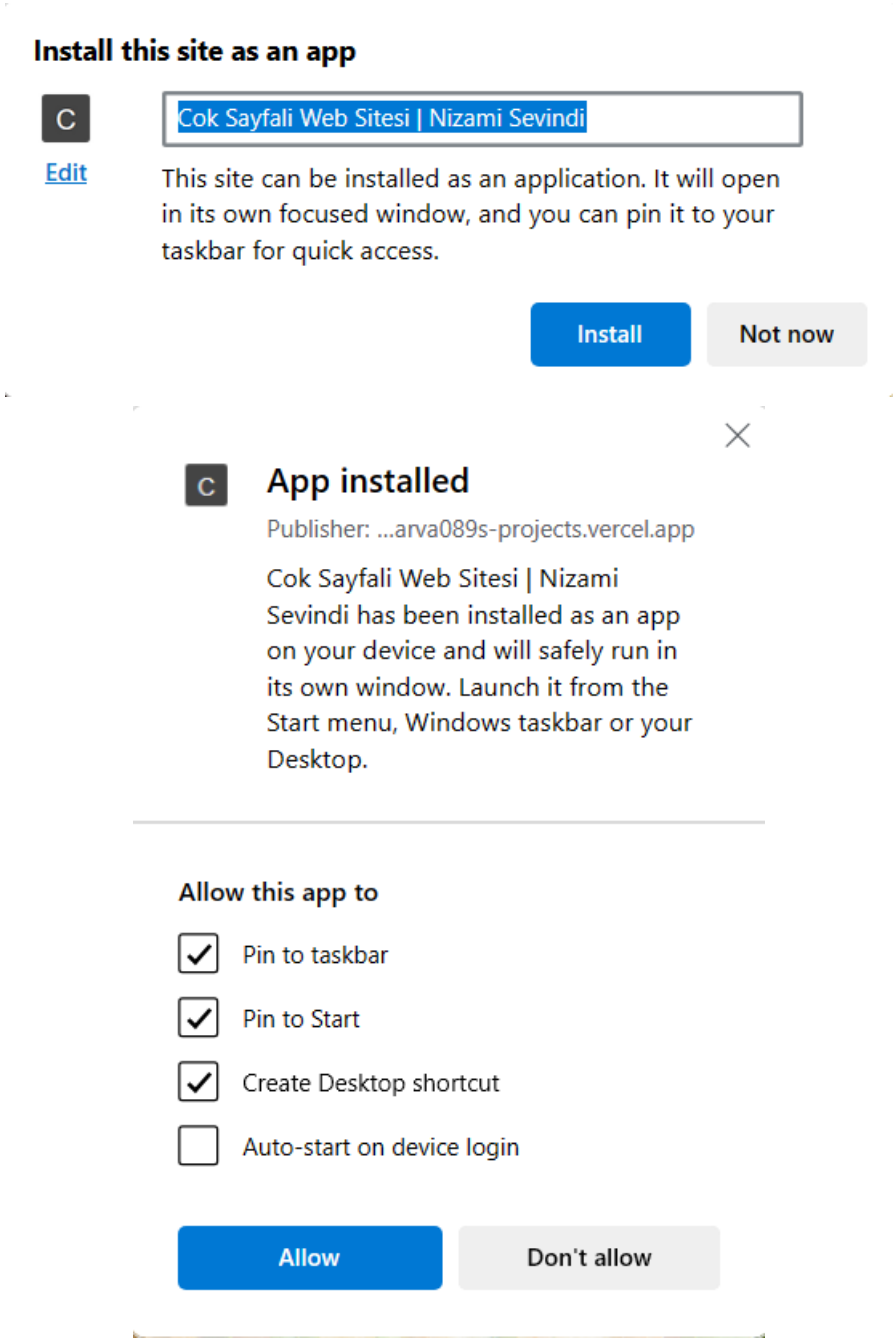
Open your hosted site on Microsoft Edge



Click on 3 dots
click on Apps
select install app option

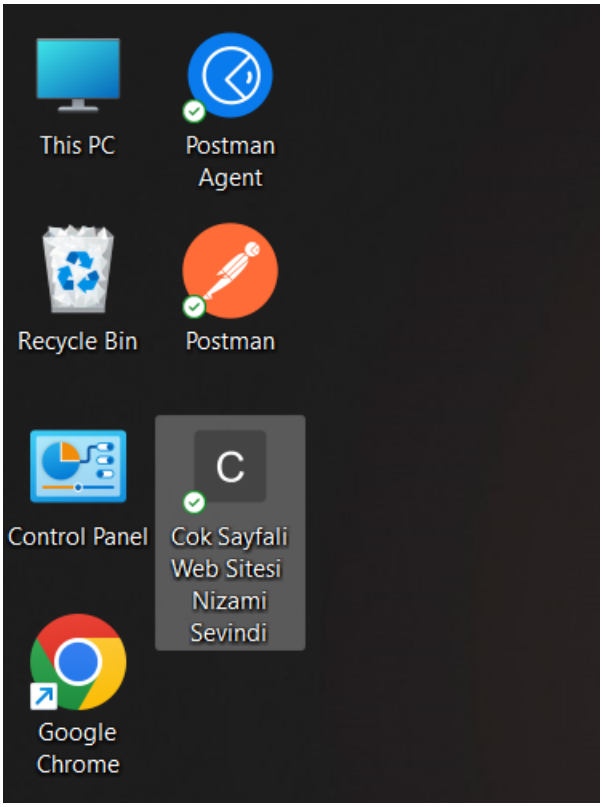


Click on Install



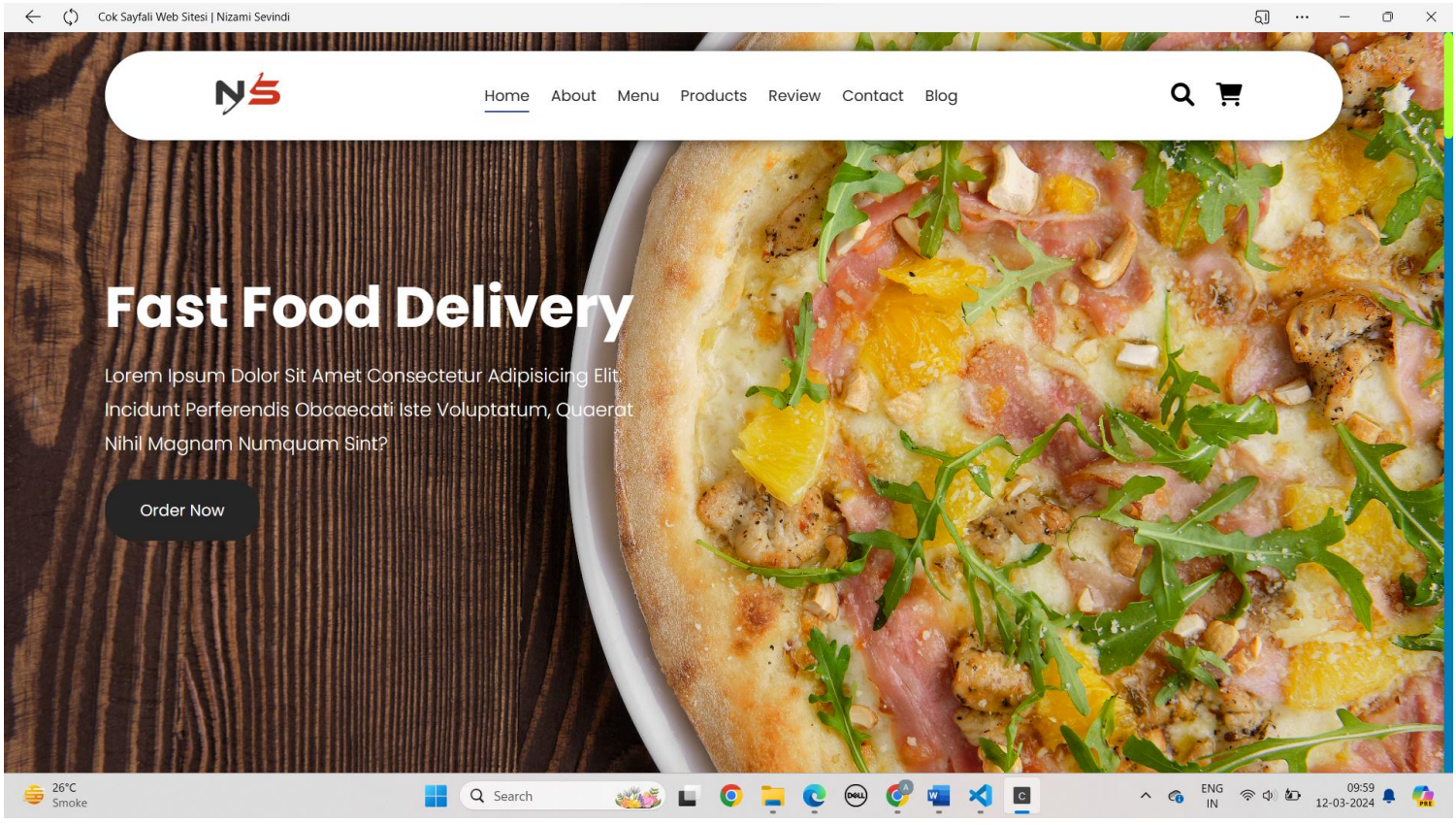
App icon will appear at the bottom

Output:



Desktop App Created Successfully.

Open Desktop App:



Conclusion:

In this experiment, we have successfully created a basic progressive web app of our web page and installed it in our desktop successfully.

MAD & PWA Lab

Journal

Experiment No.	08
Experiment Title.	To code and register a service worker, and complete the install and activation process for a new service worker for the E-commerce PWA
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

MAD and PWA Lab

Name: Atharva Chavan Class: D15A Roll no:10

Experiment – 8

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 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.

Here's a theory on how to code and register a service worker and complete the installation and activation process for a new service worker:

Service Worker Registration:

Start by creating a JavaScript file for the service worker (e.g., service-worker.js).

In your main HTML file (e.g., index.html), include code to register the service worker. This code typically resides within a <script> tag and should be placed near the bottom of the <body> tag to ensure the DOM content is fully loaded before registering the service worker.

Use the navigator.serviceWorker.register() method to register the service worker script. This method takes the path to the service worker script as its parameter.

It's a best practice to perform feature detection to check if the browser supports service workers before attempting to register one.

Service Worker Installation:

Once the service worker is registered, the browser will attempt to install it.

Inside the service worker script, listen for the install event. This event is triggered when the browser detects a new service worker for the first time.

Upon installation, you can perform tasks such as caching static assets (HTML, CSS, JavaScript, images) using the Cache API. This ensures that the application shell is available offline.

During installation, pre-cache essential assets by fetching and storing them in the cache storage.

Service Worker Activation:

After the service worker is successfully installed, it enters the activation phase.

Listen for the activate event inside the service worker script. This event is triggered once the service worker becomes active.

During activation, you can clean up old caches from previous versions of the service worker to ensure the application uses the latest assets.

Remove outdated caches using the CacheStorage API, typically by comparing cache keys to the current version and deleting obsolete caches.

Testing and Debugging:

Test the service worker functionality thoroughly, both in online and offline scenarios, to ensure proper caching and offline behavior.

Use browser developer tools to debug service worker code, inspect cache storage, and monitor service worker lifecycle events.

Consider implementing logging and error handling within the service worker to facilitate debugging in production environments.

Code:
index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <!-- CSS LINK -->
    <link rel="stylesheet" href="style.css">
    <!-- Fontawesome -->
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css" integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWaZqXgel0g==" crossorigin="anonymous" referrerpolicy="no-referrer" />
    <title>Cok Sayfali Web Sitesi | Nizami Sevindi</title>
</head>
    <!-- Google tag (gtag.js) -->
<script async src="https://www.googletagmanager.com/gtag/js?id=G-PX8CKQHBTF"></script>
<script>
  window.dataLayer = window.dataLayer || [];
```

```
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-PX8CKQHBTF');
</script>
<body>
    <!-------------------------------------------- HEADER SECTION -->
    <header class="header" >
        <a href="#" class="logo">
            <img src="./image/pngegg.png" alt="logo">
        </a>
        <nav class="navbar">
            <a href="./index.html" class="active">home</a>
            <a href="./about.html">about</a>
            <a href="./menu.html">menu</a>
            <a href="./products.html">products</a>
            <a href="./review.html">review</a>
            <a href="./contact.html">contact</a>
            <a href="./blog.html">blog</a>
        </nav>
        <div class="buttons">
            <button id="search-btn">
                <i class="fas fa-search"></i>
            </button>
            <button id="cart-btn">
                <i class="fas fa-shopping-cart"></i>
            </button>
            <button id="menu-btn">
                <i class="fas fa-bars"></i>
            </button>
        </div>
        <div class="search-form">
            <input type="text" class="search-input" id="search-box"
placeholder="Search">
            <i class="fas fa-search"></i>
        </div>
        <div class="cart-items-container">
            <div class="cart-item">
                <i class="fas fa-times"></i>
                <img src="./image/pngwing.com (1).png" alt="menu">
                <div class="content">
                    <h3>cart item 01</h3>
                    <div class="price">$15.99 </div>
                </div>
            </div>
            <div class="cart-item">
                <i class="fas fa-times"></i>
                <img src="./image/pngwing.com (2).png" alt="menu">
```

```
                <div class="content">
                    <h3>cart item 02</h3>
                    <div class="price">$16.99 </div>
                </div>
            </div>
            <div class="cart-item">
                <i class="fas fa-times"></i>
                <img src="./image/pngwing.com (3).png" alt="menu">
                <div class="content">
                    <h3>cart item 03</h3>
                    <div class="price">$13.99 </div>
                </div>
            </div>
            <div class="cart-item">
                <i class="fas fa-times"></i>
                <img src="./image/pngwing.com (4).png" alt="menu">
                <div class="content">
                    <h3>cart item 04</h3>
                    <div class="price">$12.99 </div>
                </div>
            </div>
            <a href="#" class="btn">check out </a>
        </div>
    </header>
    <!----------------------------------------------HEADER SECTION -->

    <!----------------------------------------------HOME SECTION -->
    <section class="home" id="home">
        <div class="content">
        <h3>Fast Food Delivery</h3>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt
perferendis obcaecati iste voluptatum, quaerat nihil magnam numquam sint? </p>
        <a href="#" class="btn">order now</a>
        </div>
    </section>
    <!----------------------------------------------HOME SECTION -->

    <!----------------------------------------------MENU SECTION -->
    <section class="menu" id="menu">
        <h1 class="heading">our <span>menu</span></h1>
        <div class="box-container">
            <div class="box">

                <div class="box-head">
                    <img src="./image/pngwing.com.png" alt="">
                    <span class="menu-category">Pizza</span>
                    <h3>6 Mini Pizzas</h3>
                    <div class="price">$104.99 <span>$119.99</span></div>
```

```
            </div>
            <div class="box-bottom">
                <a href="#" class="btn">add to cart</a>
            </div>
        </div>
        <div class="box">

            <div class="box-head">
                <img src="./image/pngwing.com (2).png" alt="">
                <span class="menu-category">Burger</span>
                <h3>5 Mini Burgers</h3>
                <div class="price">$99.99 <span>$105.99</span></div>
            </div>
            <div class="box-bottom">
                <a href="#" class="btn">add to cart</a>
            </div>
        </div>
        <div class="box">

            <div class="box-head">
                <img src="./image/pngwing.com (1).png" alt="">
                <span class="menu-category">Pizza</span>
                <h3>2 Mixed Pizzas</h3>
                <div class="price">$49.99 <span>$59.99</span></div>
            </div>
            <div class="box-bottom">
                <a href="#" class="btn">add to cart</a>
            </div>
        </div>
        <div class="box">

            <div class="box-head">
                <img src="./image/pngwing.com (3).png" alt="">
                <span class="menu-category">Burger</span>
                <h3>3 Meatball Burgers</h3>
                <div class="price">$79.99 <span>$99.99</span></div>
            </div>
            <div class="box-bottom">
                <a href="#" class="btn">add to cart</a>
            </div>
        </div>
    </div>
</section>
<!-------------------------------------------MENU SECTION -->


<!-------------------------------------------PRODUCTS SECTION -->
<section class="products" id="products">
    <h1 class="heading">our <span>products</span> </h1>
```

```
        <div class="box-container">
            <div class="box">
                <div class="box-head">
                    <span class="title">mini burger</span>
                    <a href="#" class="name">Bacon Burger</a>
                </div>
                <div class="image">
                    <img src="./image/pngwing.com (6).png" alt="">
                </div>
                <div class="box-bottom">
                    <div class="info">
                        <b class="price">$6.00</b>
                        <span class="amount">110gr / 300 Cal</span>
                    </div>
                    <div class="product-btn">
                        <a href="#">
                            <i class="fas fa-plus"></i>
                        </a>
                    </div>
                </div>
            </div>
            <div class="box">
                <div class="box-head">
                    <span class="title">cheese burger</span>
                    <a href="#" class="name">cheese Burger</a>
                </div>
                <div class="image">
                    <img src="./image/pngwing.com (4).png" alt="">
                </div>
                <div class="box-bottom">
                    <div class="info">
                        <b class="price">$12.00</b>
                        <span class="amount">140gr / 2500 Cal</span>
                    </div>
                    <div class="product-btn">
                        <a href="#">
                            <i class="fas fa-plus"></i>
                        </a>
                    </div>
                </div>
            </div>
            <div class="box">
                <div class="box-head">
                    <span class="title">Double burger</span>
                    <a href="#" class="name">Double Burger</a>
                </div>
                <div class="image">
                    <img src="./image/pngwing.com (5).png" alt="">
```

```
                </div>
                <div class="box-bottom">
                    <div class="info">
                        <b class="price">$24.00</b>
                        <span class="amount">440gr / 600 Cal</span>
                    </div>
                    <div class="product-btn">
                        <a href="#">
                            <i class="fas fa-plus"></i>
                        </a>
                    </div>
                </div>
            </div>
        </div>
    </section>
    <!-----------------------------------------------PRODUCTS SECTION -->

    <!-----------------------------------------------ABOUT US SECTION -->
    <section class="about" id="about">
        <h1 class="heading">about <span>us</span> </h1>

        <div class="row">
            <div class="image">
                <img src="./image/pngwing.com (7).png" alt="">
            </div>
            <div class="content">
                <h3>What is the secret receipe of our burgers</h3>
                <div class="paragraph">
                    <p>Lorem ipsum, dolor sit amet consectetur adipisicing
elit. Quia officia id et, corrupti assumenda.</p>
                    <p>Lorem ipsum, dolor sit amet consectetur adipisicing
elit. Quia officia id et, corrupti assumenda.</p>
                    <p>Lorem ipsum, dolor sit amet consectetur adipisicing
elit. Quia officia id et, corrupti assumenda.</p>
                </div>
                    <a href="#" class="btn">Learn More</a>
            </div>
        </div>
    </section>
    <!-----------------------------------------------ABOUT US SECTION -->

    <!-----------------------------------------------REVIEW SECTION -->
    <section class="review" id="review">
        <h1 class="heading">customer's <span>review</span> </h1>
        <div class="box-container">
            <div class="box">
                <img src="./image/pngwing.com (8).png" alt="quote">
```

```
                <p> Dicta totam suscipit vero praesentium excepturi facilis,
fuga at architecto dolor tempora molestias quam dignissimos sit. Molestiae
temporibus ratione quas placeat possimus!</p>
                <img src="./image/pngwing.com (9).png" alt="customer-avatar"
class="user">
                <h3>Patrick Hellinger</h3>
                <div class="stars">
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star-half-alt"></i>
                </div>
            </div>
            <div class="box">
                <img src="./image/pngwing.com (8).png" alt="quote">
                <p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
Fuga at architecto dolor tempora molestias quam dignissimos sit. Molestiae
temporibus ratione quas placeat possimus!</p>
                <img src="./image/pngwing.com (10).png" alt="customer-avatar"
class="user">
                <h3>Serena Williams</h3>
                <div class="stars">
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star-half-alt"></i>
                </div>
            </div>
            <div class="box">
                <img src="./image/pngwing.com (8).png" alt="quote">
                <p>Lorem ipsum dolor sit amet consectetur adipisicing elit.
Dicta totam suscipit vero praesentium excepturi facilis, fuga at architecto
dolor tempora molestias quam dignissimos possimus!</p>
                <img src="./image/pngwing.com (11).png" alt="customer-avatar"
class="user">
                <h3>Helen Marksen</h3>
                <div class="stars">
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star"></i>
                    <i class="fas fa-star-half-alt"></i>
                </div>
            </div>
        </div>
    </section>
```

```
    <!-------------------------------------------REVIEW SECTION -->
    <!-------------------------------------------CONTACT SECTION -->
    <section class="contact" id="contact">
        <h1 class="heading">contact <span>us</span> </h1>
        <div class="row">
            <iframe class="map"
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d983.3235940970079!2
d8.540719273659763!3d47.3713194174677!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!
3m3!1m2!1s0x47900a00aa1e1d17%3A0x278f576acdd580f5!2sStorchen%20Z%C3%BCrich%20-
%20Lifestyle%20Boutique%20Hotel!5e0!3m2!1sde!2sch!4v1658505945506!5m2!1sde!2sc
h" allowfullscreen="" loading="lazy" referrerpolicy="no-referrer-when-
downgrade"></iframe>
            <form>
                <h3>get in touch</h3>
                <div class="inputBox">
                    <i class="fas fa-user"></i>
                    <input type="text" placeholder="name">
                </div>
                <div class="inputBox">
                    <i class="fas fa-envelope"></i>
                    <input type="email" placeholder="email">
                </div>
                <div class="inputBox">
                    <i class="fas fa-phone"></i>
                    <input type="number" placeholder="number">
                </div>
                <input type="submit" class="btn" value="contact now">
            </form>
        </div>
    </section>
    <!-------------------------------------------CONTACT SECTION -->

    <!-------------------------------------------BLOG SECTION -->
    <section class="blog" id="blog">
        <h1 class="heading">our <span>blog</span> </h1>
        <div class="box-container">
            <div class="box-full">
                <div class="image">
                    <img src="./image/delicious-burger-on-wooden-board-2022-
03-04-05-58-25-utc.jpg" alt="">
                </div>
                <div class="content">
                    <a href="#" class="title">how to make burgers</a>
                    <span>by admin / 10st may, 2020</span>
                    <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit. Minus eos esse nesciunt cupiditate expedita.</p>
                    <a href="#" class="btn">read more</a>
                </div>
```

```
            </div>
            <div class="box-full">
                <div class="image">
                    <img src="./image/delicious-tasty-burgers-on-wooden-
background-2021-08-26-15-25-13-utc.jpg" alt="">
                </div>
                <div class="content">
                    <a href="#" class="title">how to make burgers</a>
                    <span>by admin / 10st may, 2020</span>
                    <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit. Minus eos esse nesciunt cupiditate expedita.</p>
                    <a href="#" class="btn">read more</a>
                </div>
            </div>
            <div class="box-full">
                <div class="image">
                    <img src="./image/handmade-burger-on-dark-background-
delicious-blac-2021-10-21-02-27-27-utc.jpg" alt="">
                </div>
                <div class="content">
                    <a href="#" class="title">how to make burgers</a>
                    <span>by admin / 10st may, 2020</span>
                    <p>Lorem ipsum dolor sit amet, consectetur adipisicing
elit. Minus eos esse nesciunt cupiditate expedita.</p>
                    <a href="#" class="btn">read more</a>
                </div>
            </div>
        </div>
    </section>
    <!-----------------------------------------BLOG SECTION -->

    <!-----------------------------------------FOOTER SECTION -->
    <section class="footer">
        <div class="search">
            <input type="text" class="search-input" placeholder="Search">
            <button class="btn btn-primary">search</button>
        </div>
        <div class="share">
            <a href="#" class="fab fa-facebook"></a>
            <a href="#" class="fab fa-twitter"></a>
            <a href="#" class="fab fa-instagram"></a>
            <a href="#" class="fab fa-linkedin"></a>
            <a href="#" class="fab fa-pinterest"></a>
        </div>
        <div class="links">
            <a href="#home">home</a>
            <a href="#about">about</a>
            <a href="#menu">menu</a>
```

```
            <a href="#products">products</a>
            <a href="#review">review</a>
            <a href="#contact">contact</a>
            <a href="#blog">blog</a>
        </div>
        <div class="credit">
            creaated by <span>Nizami Sevindi</span> | all rights reserved!
        </div>
    </section>
    <!-----------------------------------------------FOOTER SECTION -->

<script src="./script.js"></script>
</body>
</html>
```

main.css:

main.css:

```
* {

    margin: 0;
    padding: 0;
}

.navbar {
    display: flex;
    align-items: center;
    justify-content: center;
    position: sticky;
    top: 0;
    padding: 15px;
    cursor: pointer;
}

.background {
    background: black;
    background-blend-mode: darken;
    background-size: cover;
}

.nav-list {
    width: 70%;
    display: flex;
    align-items: center;
    gap: 20px;
    list-style: none;
```

```
}

.logo {
    display: flex;
    justify-content: center;
    align-items: center;
}

.logo img {
    width: 180px;
    border-radius: 50px;
}

.nav-list li {
    list-style: none;
    padding: 26px 30px;
    padding: 10px;
}

.nav-list li a {
    text-decoration: none;
    color: white;
}

.nav-list li a:hover {
    color: grey;
}

.rightnav {
    width: 30%;
    text-align: right;
}

#search {
    padding: 5px;
    font-size: 17px;
    border: 2px solid grey;
    border-radius: 9px;
}

.firstsection {
    background-color: green;
    height: 400px;
}

.secondsection {
    background-color: blue;
```

```
    height: 400px;
}

.box-main {
    display: flex;
    justify-content: center;
    align-items: center;
    color: black;
    max-width: 80%;
    margin: auto;
    height: 80%;
}

.firsthalf {
    width: 100%;
    display: flex;
    flex-direction: column;
    justify-content: center;
}

.secondhalf {
    width: 30%;
}

.secondhalf img {
    width: 70%;
    border: 4px solid white;
    border-radius: 150px;
    display: block;
    margin: auto;
}

.text-big {
    font-family: 'Piazzolla', serif;
    font-weight: bold;
    font-size: 35px;
}

.text-small {
    font-size: 18px;
}

.btn {
    padding: 8px 20px;
    margin: 7px 0;
    border: 2px solid white;
    border-radius: 8px;
```

```
    background: none;
    color: white;
    cursor: pointer;
}

.btn-sm {
    padding: 6px 10px;
    vertical-align: middle;
}

.section {
    height: 400px;
    display: flex;
    align-items: center;
    justify-content: center;
    max-width: 90%;
    margin: auto;
}

.section-Left {
    flex-direction: row-reverse;
}

.paras {
    padding: 0px 65px;
}

.thumbnail img {
    width: 250px;
    border: 2px solid black;
    border-radius: 26px;
    margin-top: 19px;
}

.center {
    text-align: center;
}

.text-footer {
    text-align: center;
    padding: 30px 0;
    font-family: 'Ubuntu', sans-serif;
    display: flex;
    justify-content: center;
    color: white;
}
```

```css
footer {
    text-align: center;
    padding: 15px;
}


.rightnav {
    width: 100%;
    text-align: right;
    margin-top: 10px;
}

#search {
    box-sizing: border-box;
    width: 70%;
    padding: 8px;
    font-size: 17px;
    border: 2px solid grey;
    border-radius: 9px;
}

.btn-sm {
    padding: 8px 20px;
    margin: 7px 5px;
}

img {
    max-width: 100%;
    height: auto;
}
```

App.js:

```js
if ('serviceWorker' in navigator) {

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

Service-worker.js:

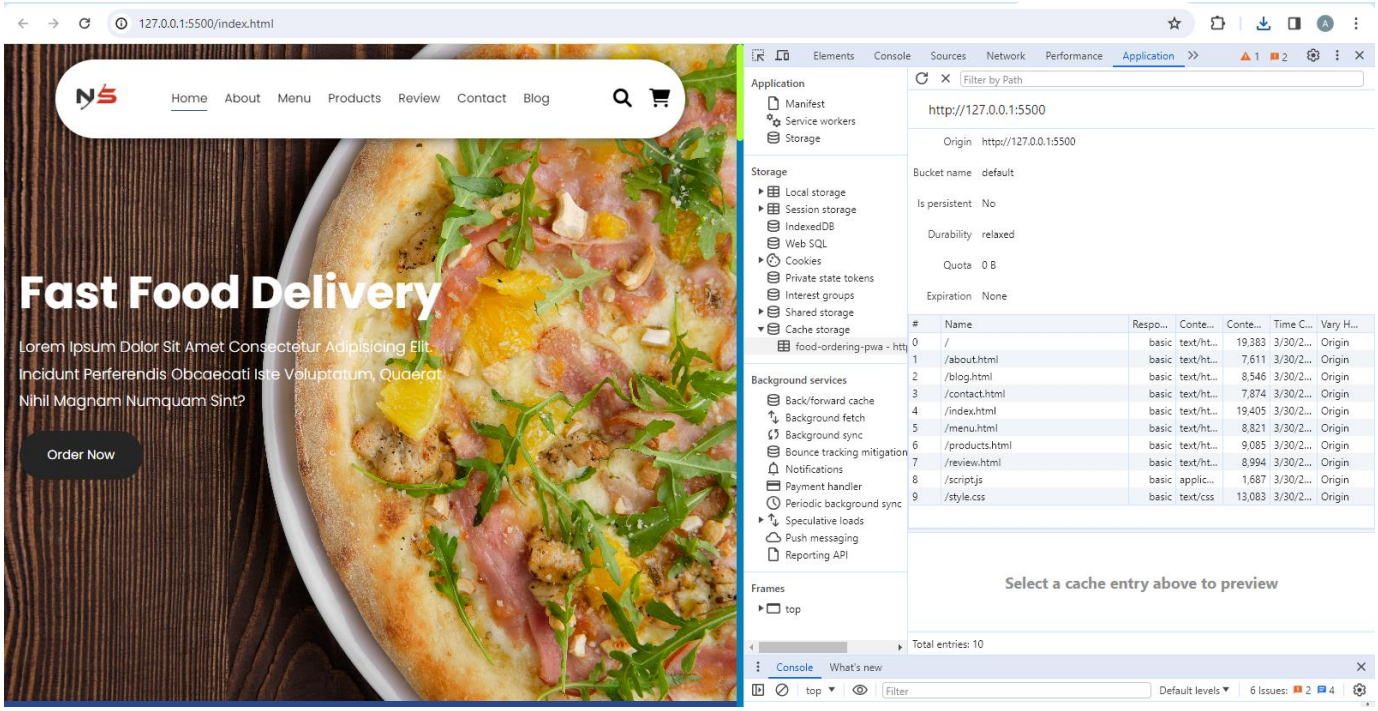
```
// service-worker.js


const cacheName = 'gfg-pwa-v1';
const assetsToCache = [
    '/',
    '/index.html',
    '/main.css',
    '/app.js'
    // Add more files and assets here as needed
];


self.addEventListener('install', event => {
    event.waitUntil(
        caches.open(cacheName)
            .then(cache => {
                return cache.addAll(assetsToCache);
            })
    );
});


self.addEventListener('activate', event => {
    event.waitUntil(
        caches.keys().then(cacheNames => {
            return Promise.all(
                cacheNames.filter(name => {
                    return name !== cacheName;
                }).map(name => {
                    return caches.delete(name);
                })
            );
        })
    );
});
```

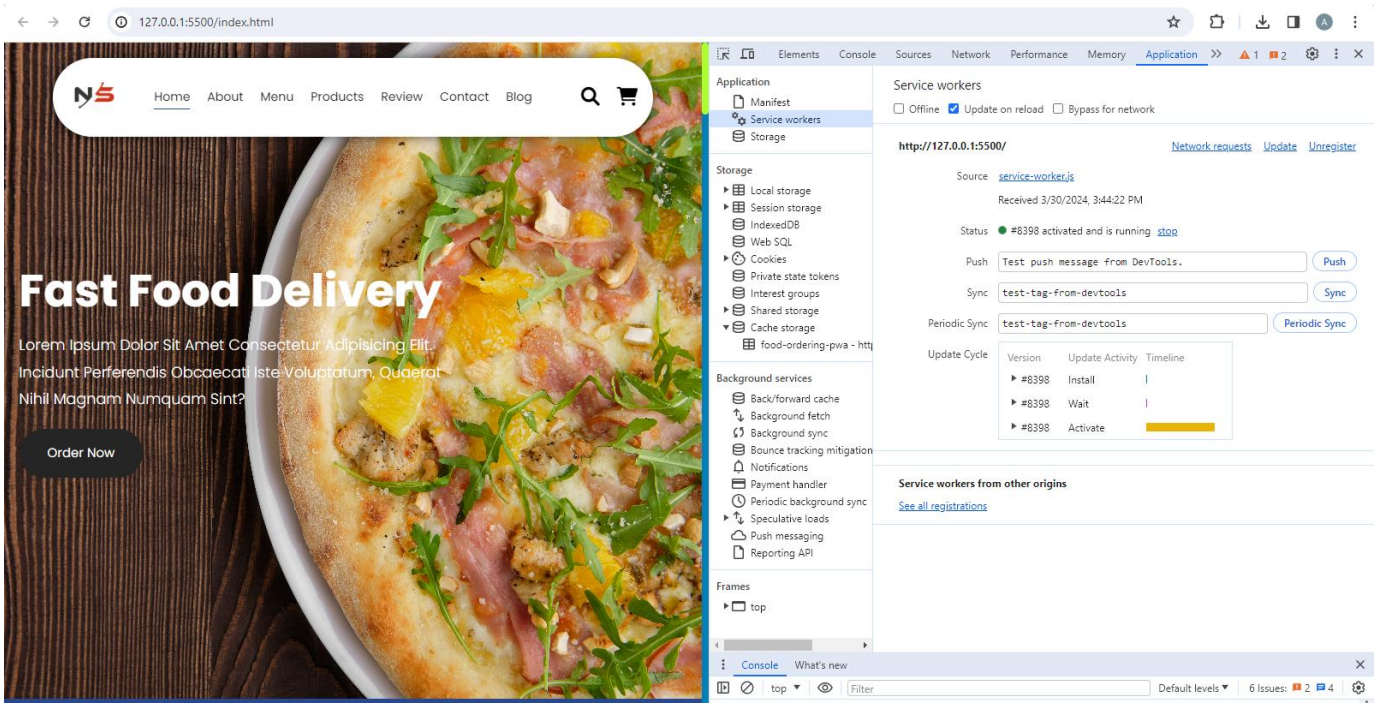
Steps for Execution:-

- Create a folder and put all 4 files main.css , service-worker.js, app.js, index.html
- Open visual studio

- Install extension Live server
- Open folder in visual studio open index.html
- On bottom right corner click go Live
- It will open html page in browser
- Go to developer tools

Output:





Conclusion:

In this experiment, we have registered a service worker, and completed the install and activation process for a new service worker for the PWA.

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.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

MAD AND PWA LAB

Name: Atharva Chavan Class: D15A Roll no: 10

Experiment – 9

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

Theory:

Service Worker

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

Things to note about Service Worker:

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

Fetch Event

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

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

- CacheFirst - In this function, if the received request has cached before, the cached response is returned to the page. But if not, a new response requested from the network.

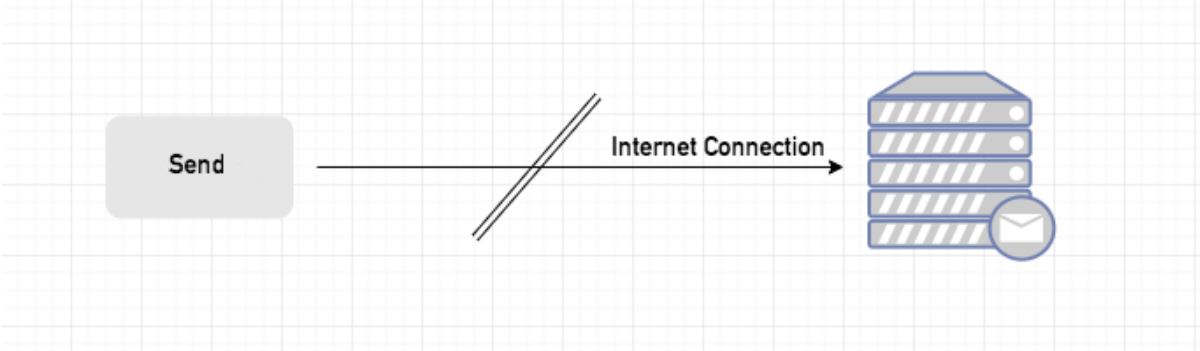
- NetworkFirst - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned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

Sync Event

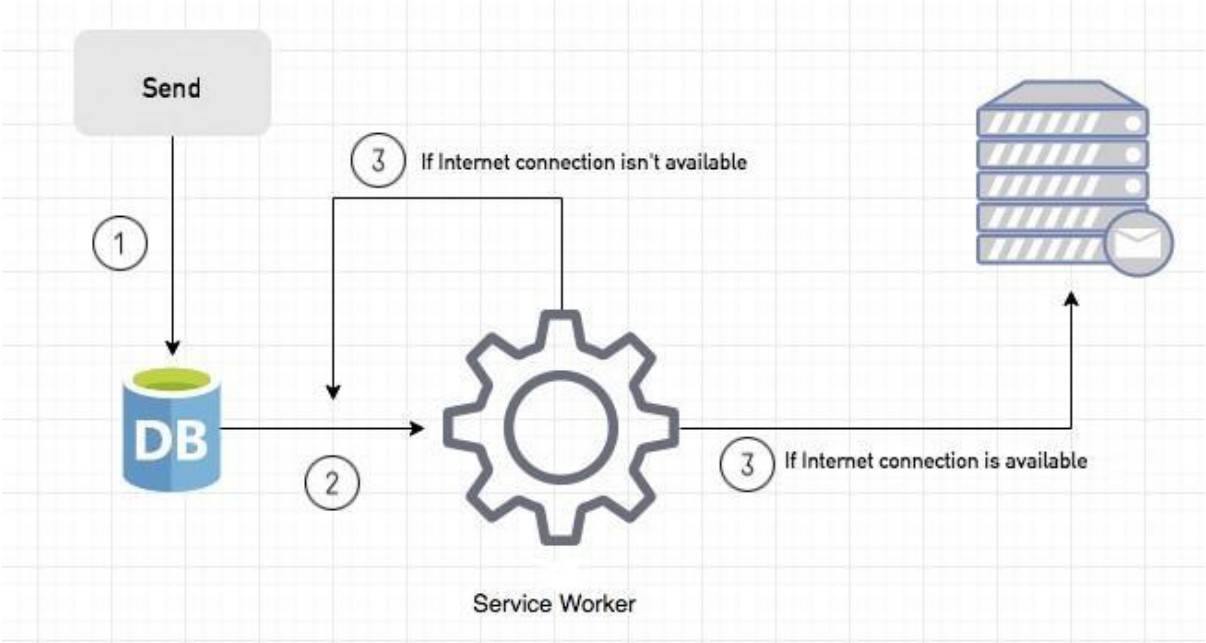
Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are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

Here is a job for the Background Sync.

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



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



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

   If the Internet connection is unavailable, the service worker waits until the connection is available even though the window is closed. When it is available, email content will be sent to Mail Server.

   You can see the working process within the following code block.

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:

sw.js

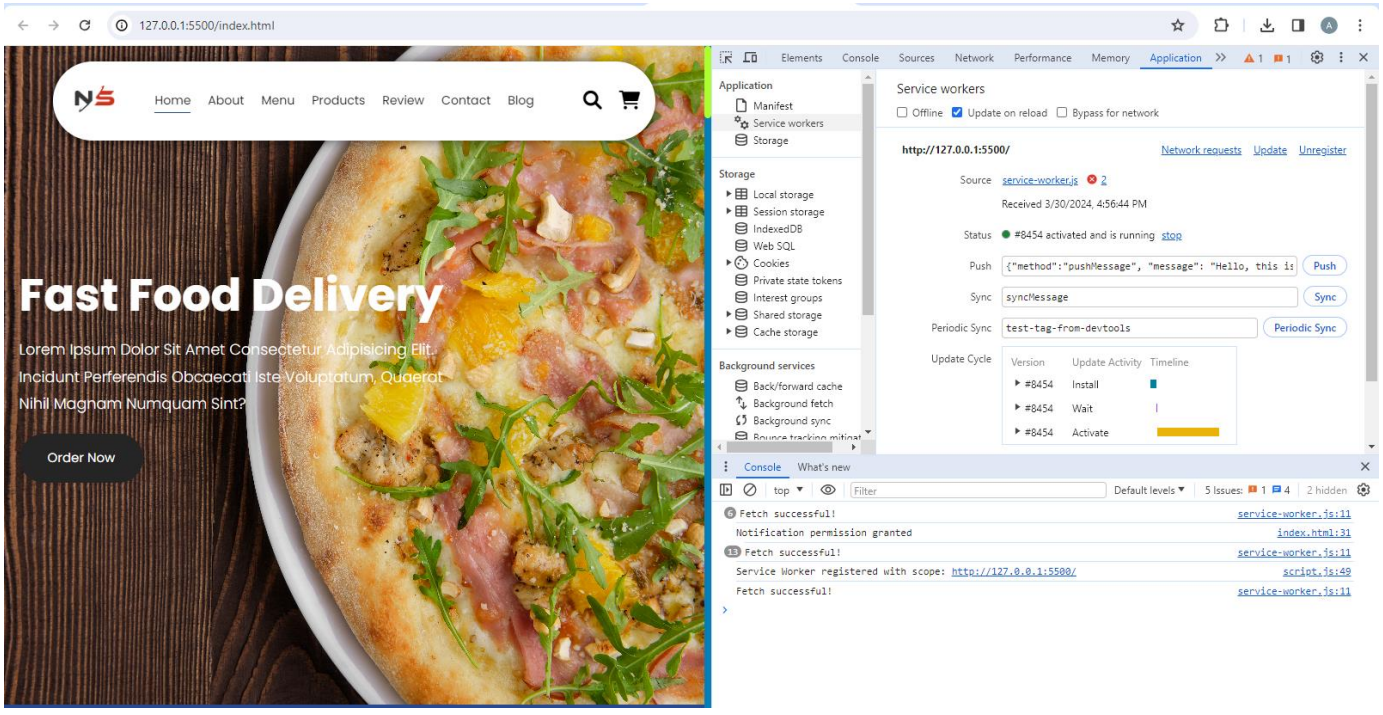
```
self.addEventListener("install", function (event) {
  event.waitUntil(preLoad());
});
self.addEventListener("fetch", function (event) {
  event.respondWith(
    checkResponse(event.request).catch(function () {
      console.log("Fetch from cache successful!");
      return returnFromCache(event.request);
    })
  );
  console.log("Fetch successful!");
  event.waitUntil(addToCache(event.requesst));
});
self.addEventListener("sync", (event) => {
  if (event.tag === "syncMessage") {
    console.log("Sync successful!");
  }
});
self.addEventListener("push", function (event) {
  if (event && event.data) {
    try {
      var data = event.data.json();
```

```
      if (data && data.method === "pushMessage") {
        console.log("Push notification sent");
        self.registration.showNotification("Twiggy", {
          body: data.message,
        });
      }
    } catch (error) {
      console.error("Error parsing push data:", error);
    }
  }
});
var preLoad = function () {
  return caches.open("offline").then(function (cache) {
    // caching index and important routes
    return cache.addAll([
      "/",
      "/index.html",
      "/style.css",
      "/app.js",
      "/blog.html",
      "/about.html",
      "/contact.html",
      "/menu.html",
      "/products.html",
      "/review.html",
      "/script.js",
    ]);
  });
};
var checkResponse = function (request) {
  return new Promise(function (fulfill, reject) {
    fetch(request)
      .then(function (response) {
        if (response.status !== 404) {
          fulfill(response);
        } else {
          reject(new Error("Response not found"));
        }
      })
      .catch(function (error) {
        reject(error);
      });
  });
};
var returnFromCache = function (request) {
  return caches.open("offline").then(function (cache) {
    return cache.match(request).then(function (matching) {
      if (!matching || matching.status == 404) {
```

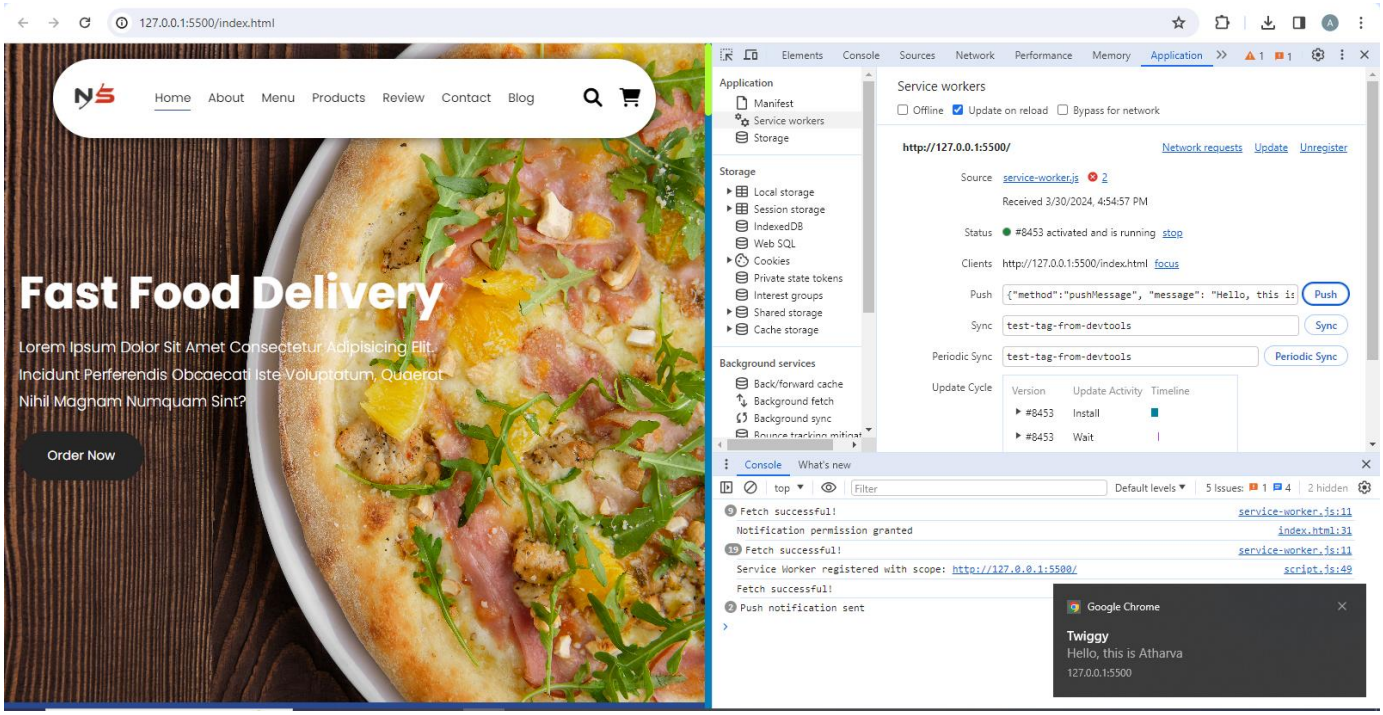
```
        return cache.match("offline.html");
      } else {
        return matching;
      }
    });
  });
};
var addToCache = function (request) {
  return caches.open("offline").then(function (cache) {
    return fetch(request).then(function (response) {
      return cache.put(request, response.clone()).then(function () {
        return response;
      });
    });
  });
};
```

Output:

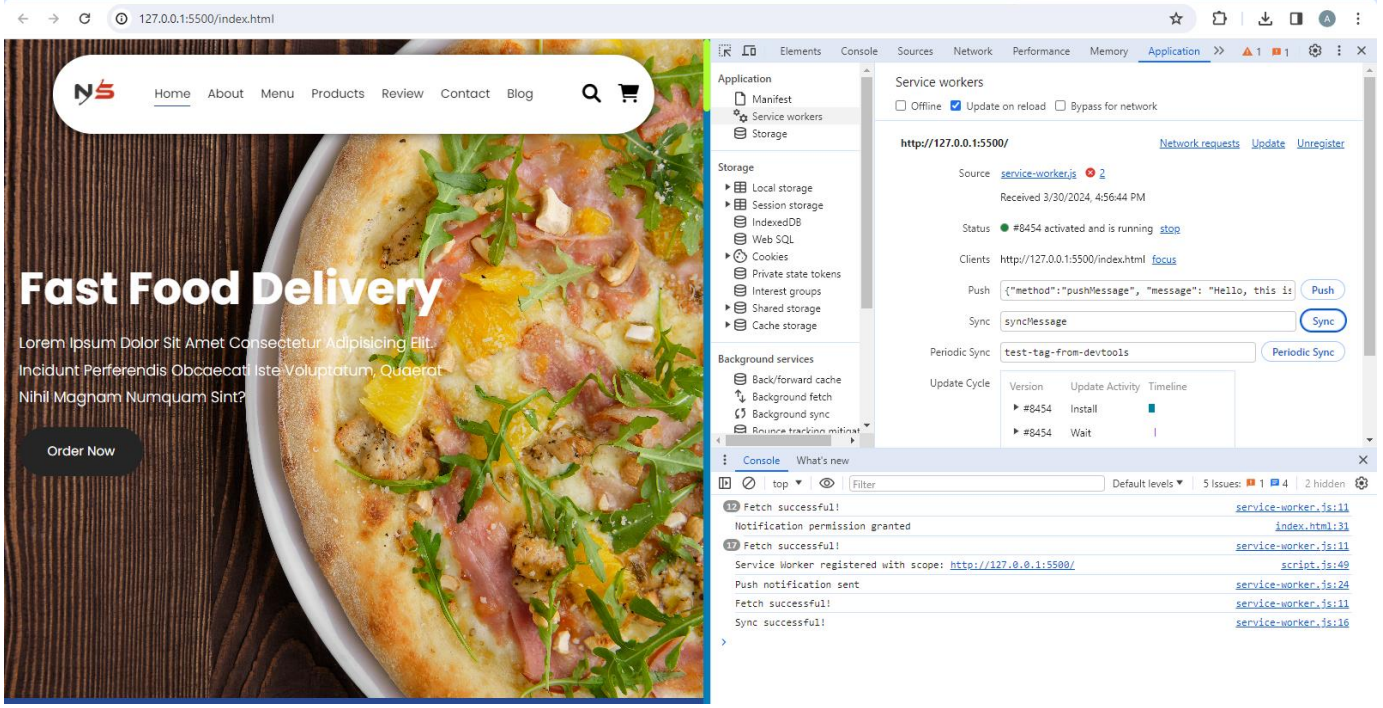
Fetch Event



Push Event



Sync Event



Conclusion:

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

MAD & PWA Lab

Journal

Experiment No.	10
Experiment Title.	To study and implement deployment of Ecommerce PWA to GitHub Pages.
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO5: Design and Develop a responsive User Interface by applying PWA Design techniques
Grade:	15

MAD and PWA Lab

Name: Atharva Chavan Class: D15A Roll no:10

Experiment - 10

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

Theory:

GitHub Pages

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

GitHub Pages provides the following key features:

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

Reasons for favoring this over Firebase:

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

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

Pros

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

Cons

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

Firebase

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

Some of the features offered by Firebase are:

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

Reasons for favoring over GitHub Pages:

1. Realtime backend made easy
2. Fast and responsive

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

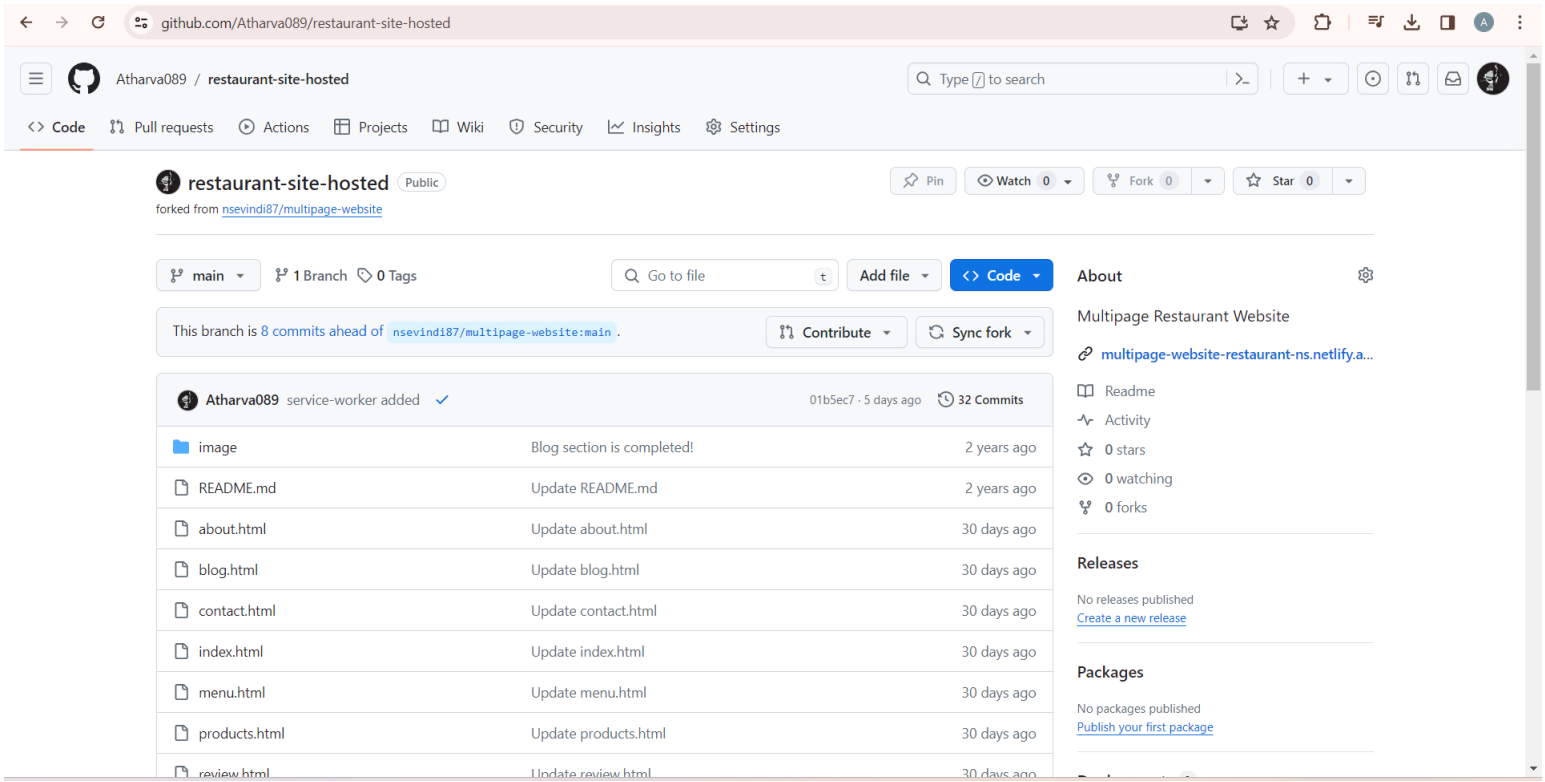
Pros

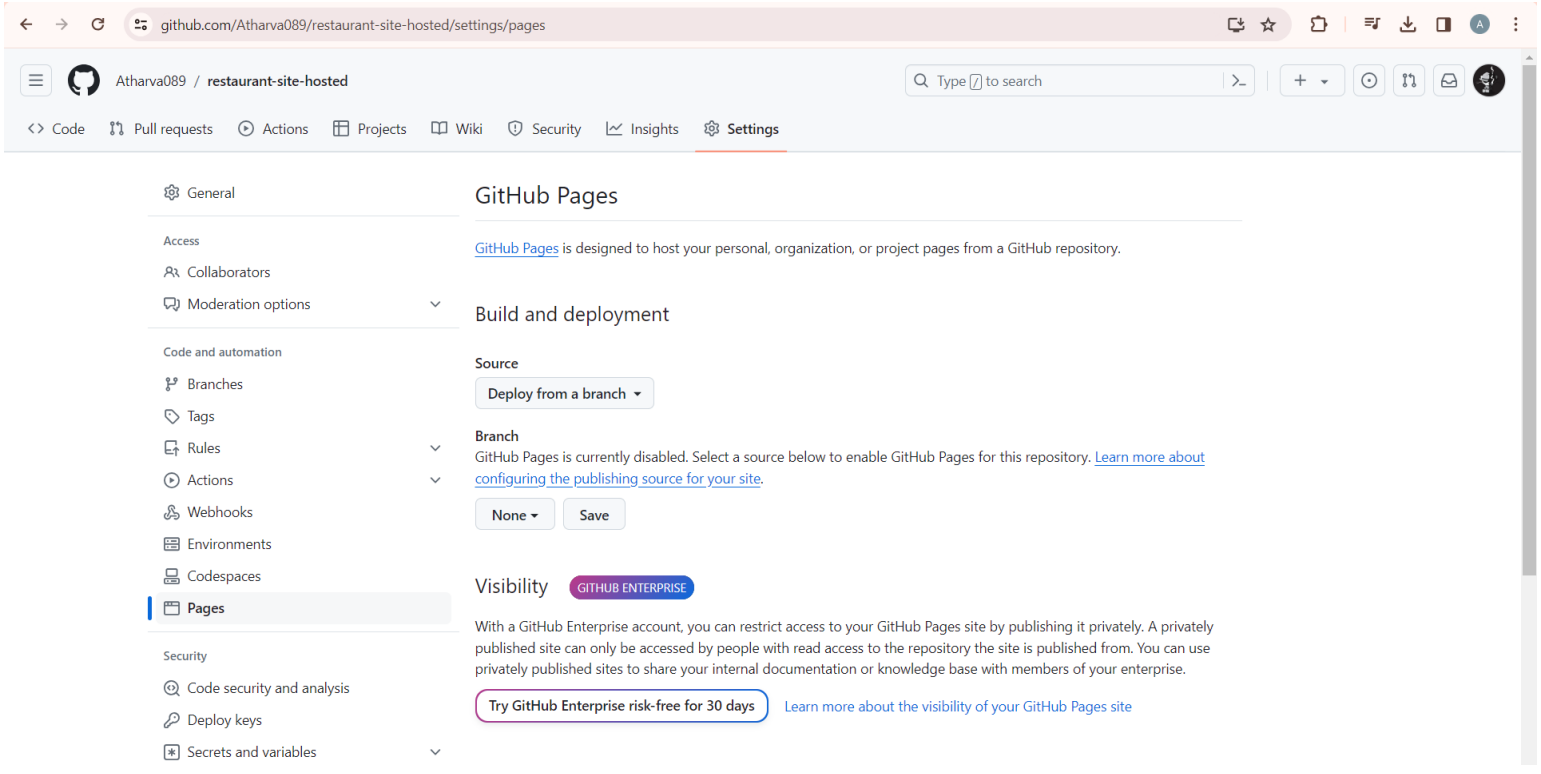
1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5.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

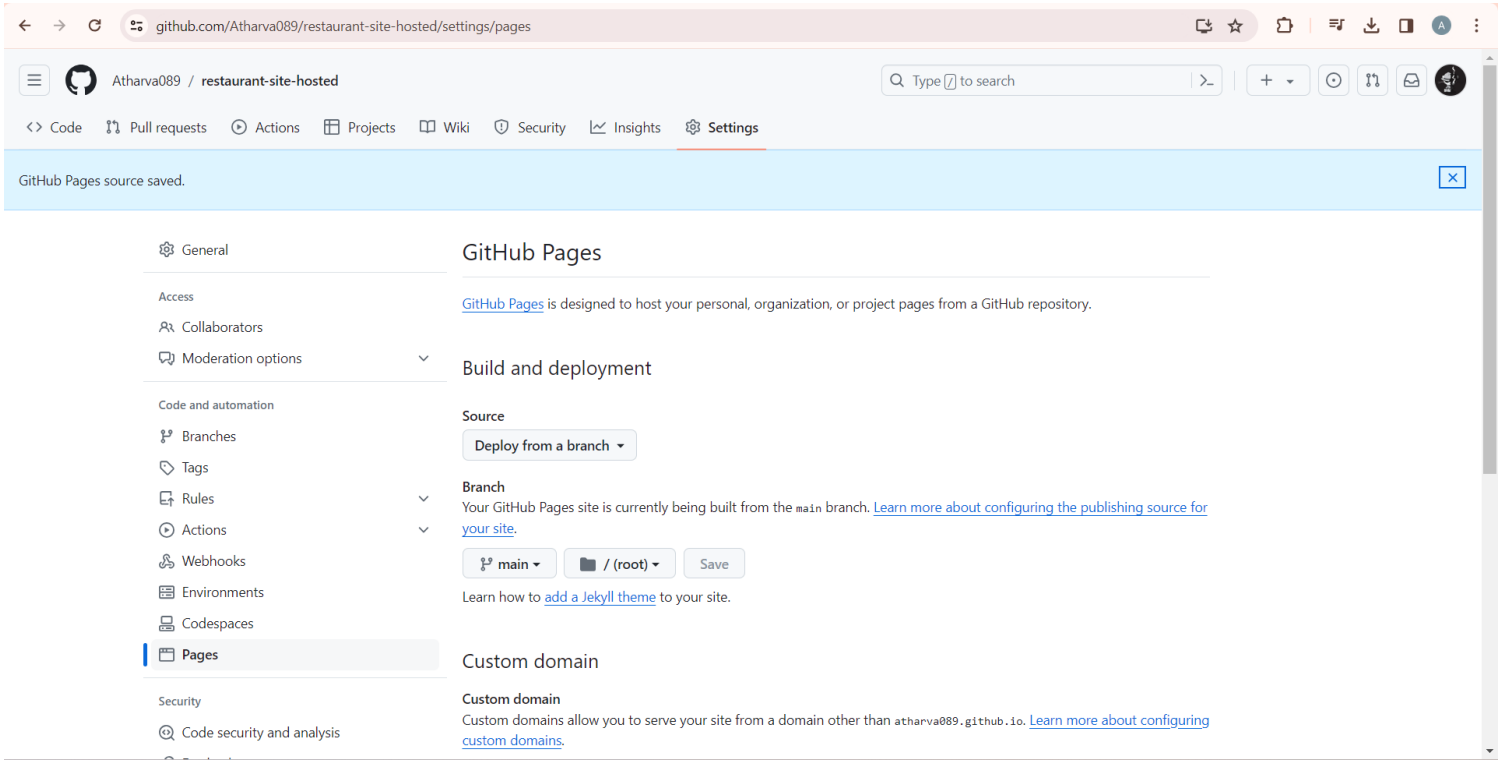
Cons

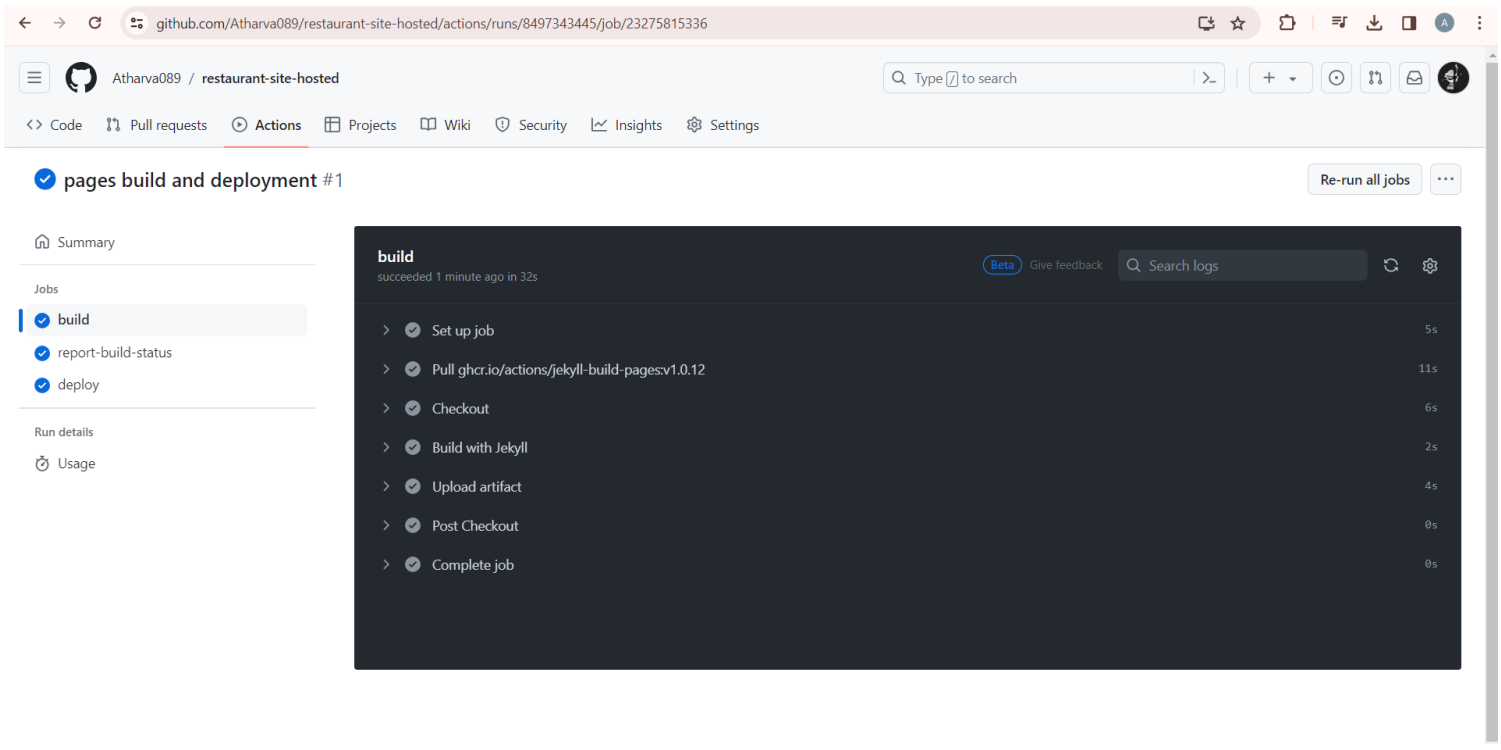
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.

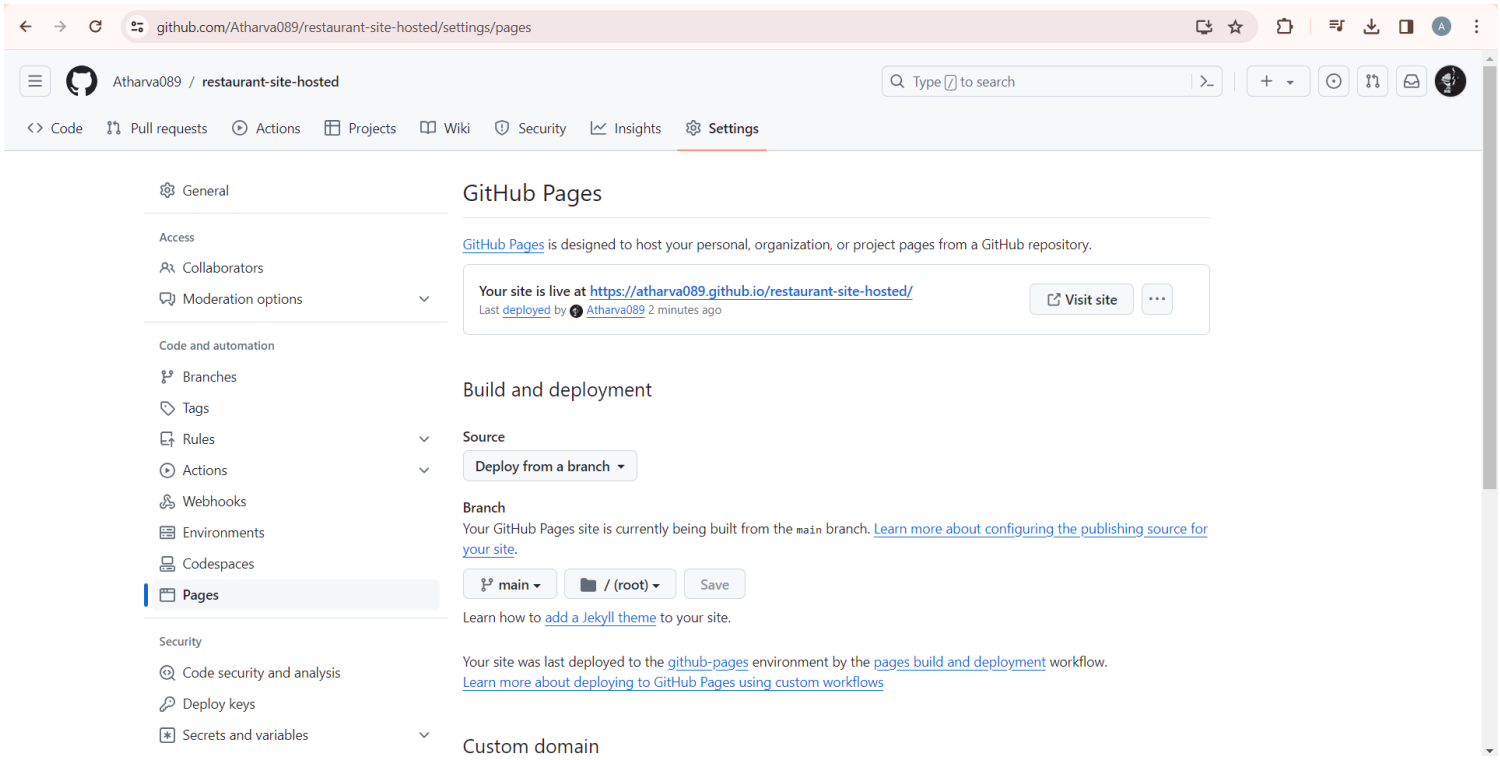
Implementation:

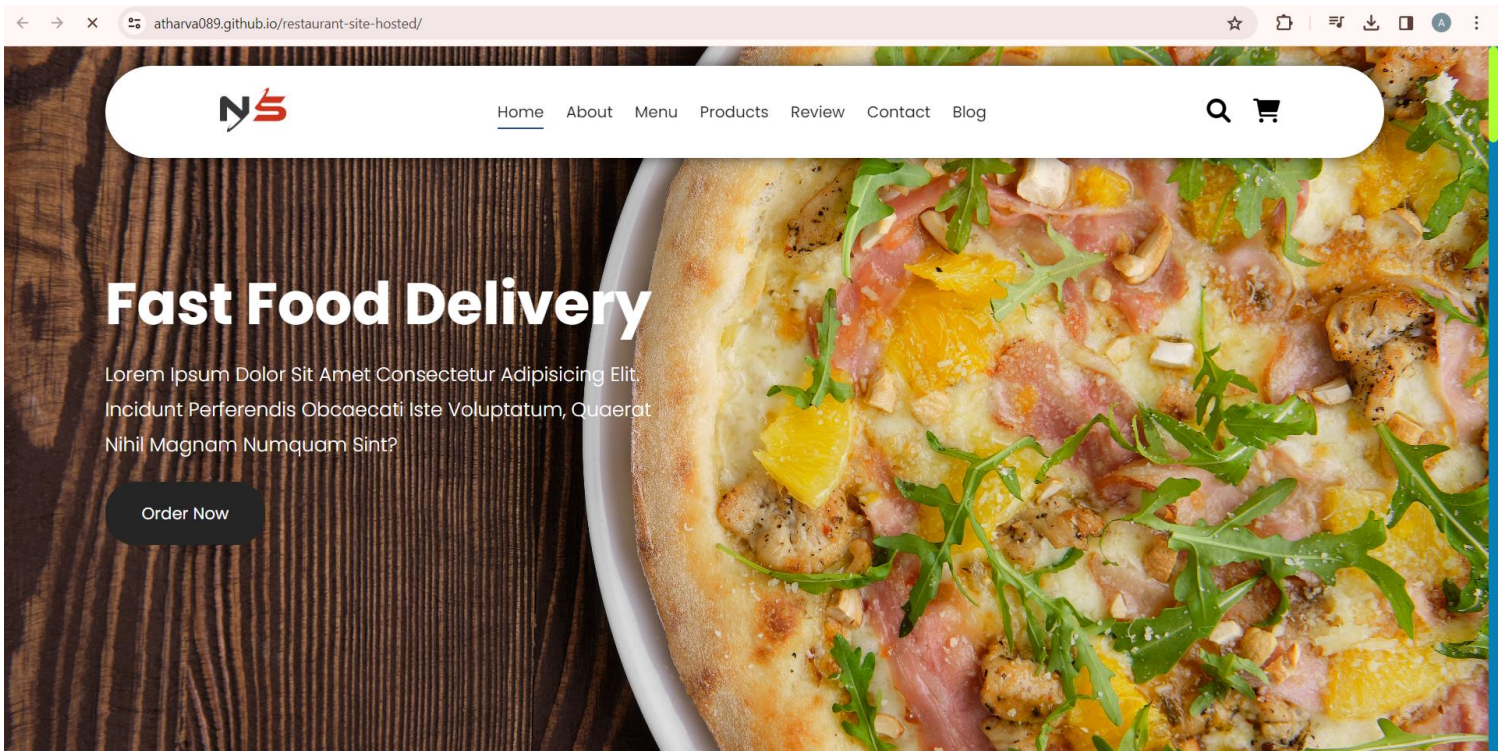





github.com/Atharva089/restaurant-site-hosted/settings/pages
Atharva089 / restaurant-site-hosted
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
GitHub Pages source saved.
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages
Security
Code security and analysis
GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
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.
Custom domain
Custom domain
Custom domains allow you to serve your site from a domain other than atharva089.github.io. Learn more about configuring custom domains.


github.com/Atharva089/restaurant-site-hosted/actions/runs/8497343445/job/23275815336
Atharva089 / restaurant-site-hosted
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
pages build and deployment #1
Re-run all jobs
Summary
Jobs
build
report-build-status
deploy
Run details
Usage
build
succeeded 1 minute ago in 32s
Beta
Give feedback
Search logs
Set up job 5s
Pull ghcr.io/actions/jekyll-build-pages:v1.0.12 11s
Checkout 6s
Build with Jekyll 2s
Upload artifact 4s
Post Checkout 0s
Complete job 0s


github.com/Atharva089/restaurant-site-hosted/settings/pages
Atharva089 / restaurant-site-hosted
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages
Security
Code security and analysis
Deploy keys
Secrets and variables
GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
Your site is live at https://atharva089.github.io/restaurant-site-hosted/
Last deployed by Atharva089 2 minutes ago
Visit site
Build and deployment
Source
Deploy from a branch
Branch
Your GitHub Pages site is currently being built from the main branch. Learn more about configuring the publishing source for your site.
main
/ (root)
Save
Learn how to add a Jekyll theme to your site.
Your site was last deployed to the github-pages environment by the pages build and deployment workflow.
Learn more about deploying to GitHub Pages using custom workflows
Custom domain


atharva089.github.io/restaurant-site-hosted/
Home
About
Menu
Products
Review
Contact
Blog
Fast Food Delivery
Lorem Ipsum Dolor Sit Amet Consectetur Adipisicing Elit.
Incidunt Perferendis Obcaecati Iste Voluptatum, Quaerat
Nihil Magnam Numquam Sint?
Order Now

Link to the Github repository:
Github link

Hosted Link: https://atharva089.github.io/restaurant-site-hosted/

Conclusion:

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

MAD & PWA Lab

Journal

Experiment No.	11
Experiment Title.	To use google Lighthouse PWA Analysis Tool to test the PWA functioning.
Roll No.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO6: Develop and Analyze PWA Features and deploy it over app hosting solution
Grade:	15

MAD and PWA Lab

Name: Atharva Chavan **Class: D15A** **Roll no:10**

Experiment – 11

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:

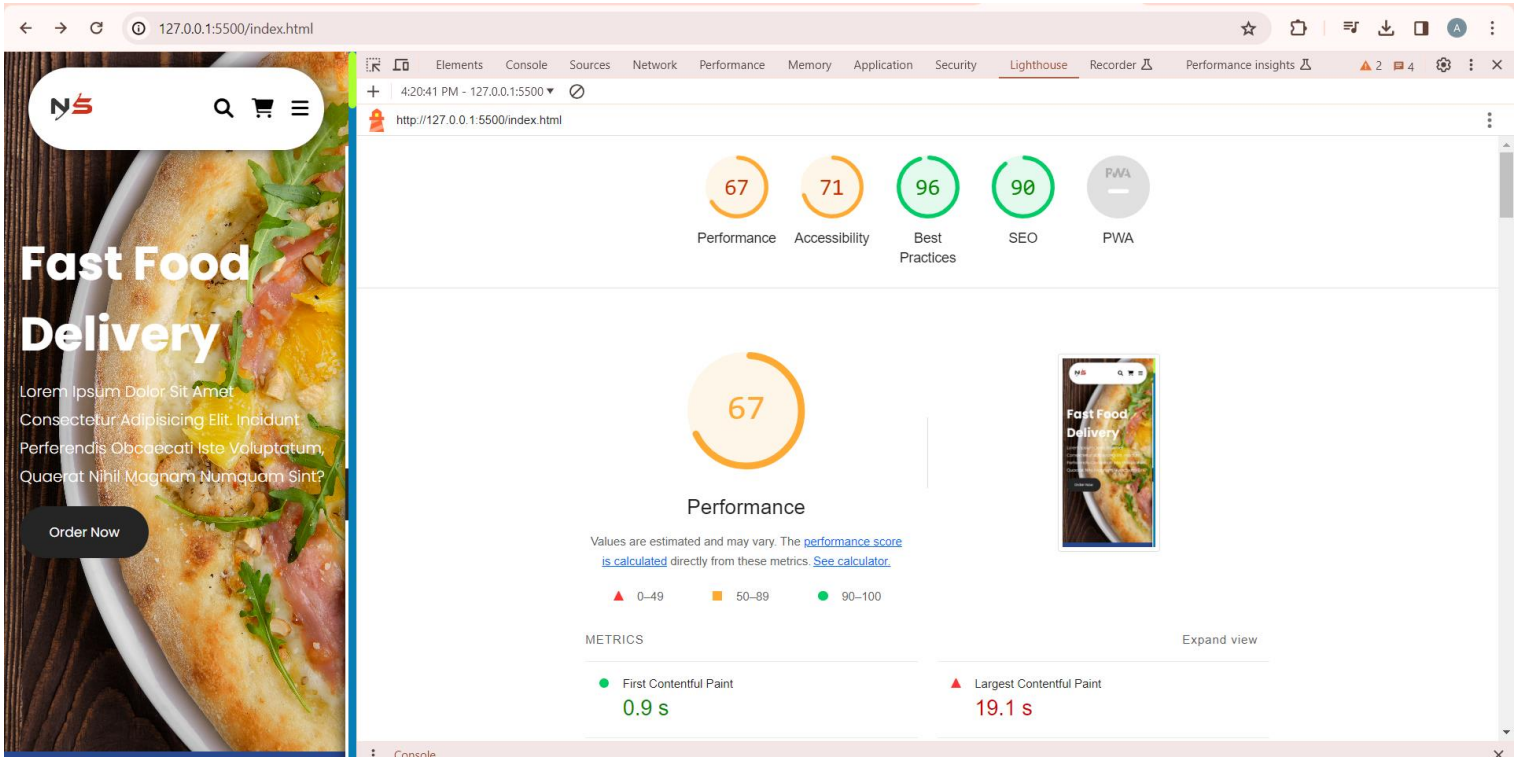
1. **Performance:** This score is an aggregation of how the page fared in aspects such as (but not limited to) loading speed, time taken for loading for basic frame(s), displaying meaningful content to the user, etc. To a layman, this score is indicative of how decently the site performs, with a score 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 applications are moving towards the PWA paradigm, where the objective is to make the application behave as close as possible to native mobile applications. Scoring points are based on the Baseline PWA checklist laid down by Google which includes Service Worker implementation(s), viewport handling, offline functionality, performance in script-disabled environments, etc.

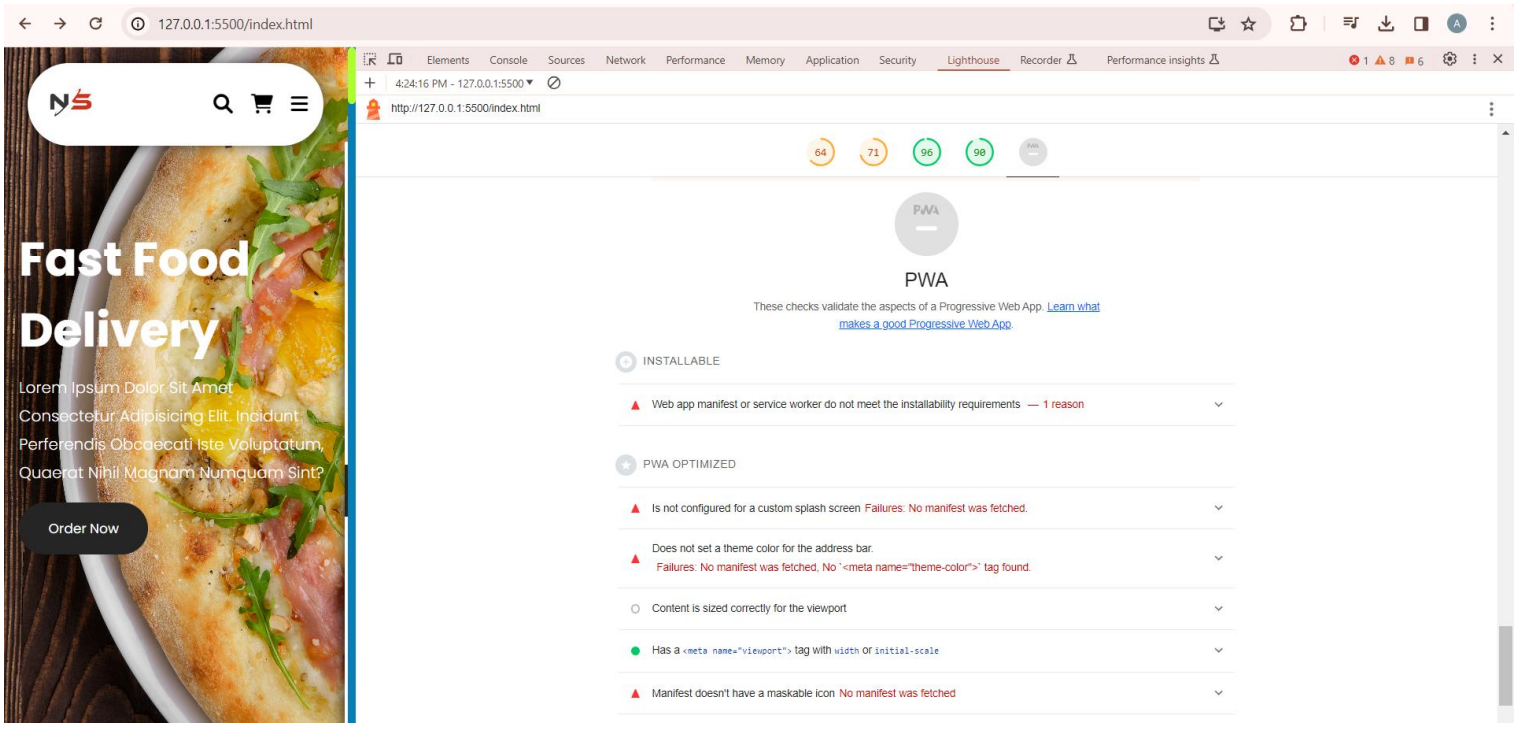
3. **Accessibility:** As you might have guessed, this metric is a measure of how accessible your website is, across a plethora of accessibility features that can be implemented in your page (such as the ‘aria-’ attributes like aria-required, audio captions, button names, etc.). Unlike the other metrics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

Code & Output:

manifest.json:

```
{
 "name": "Fast Food Delivery",
 "start_url": "index.html",
 "display": "standalone",
 "background_color": "#5900b3",
 "theme_color": "black",
 "scope": ".",
 "description": "This is a Fast food delivery app.",
 "icons": [
  {
   "src": "image/cropped.jpg",
   "sizes": "192x192",
   "type": "image/png"
  },
  {
   "src": "image/CompressJPEG.online_512x512_image.jpg",
   "sizes": "512x512",
   "type": "image/png"
  }
 ]
}
```





After Changes made to the manifest.json :

```
{
 "name":"Fast Food Delivery",
 "start_url":"index.html",
 "display":"standalone",
 "background_color":"#5900b3",
```

```
"theme_color":"black",
"scope": ".",
"description":"This is a Fast food delivery app.",
"icons":[
{
"src":"image/cropped.jpg",
"sizes":"192x192",
"type":"image/png",
"purpose":"any maskable"
},
{
"src":"image/CompressJPEG.online_512x512_image.jpg",
"sizes":"512x512",
"type":"image/png",
"purpose":"any maskable"
}
]
}
```

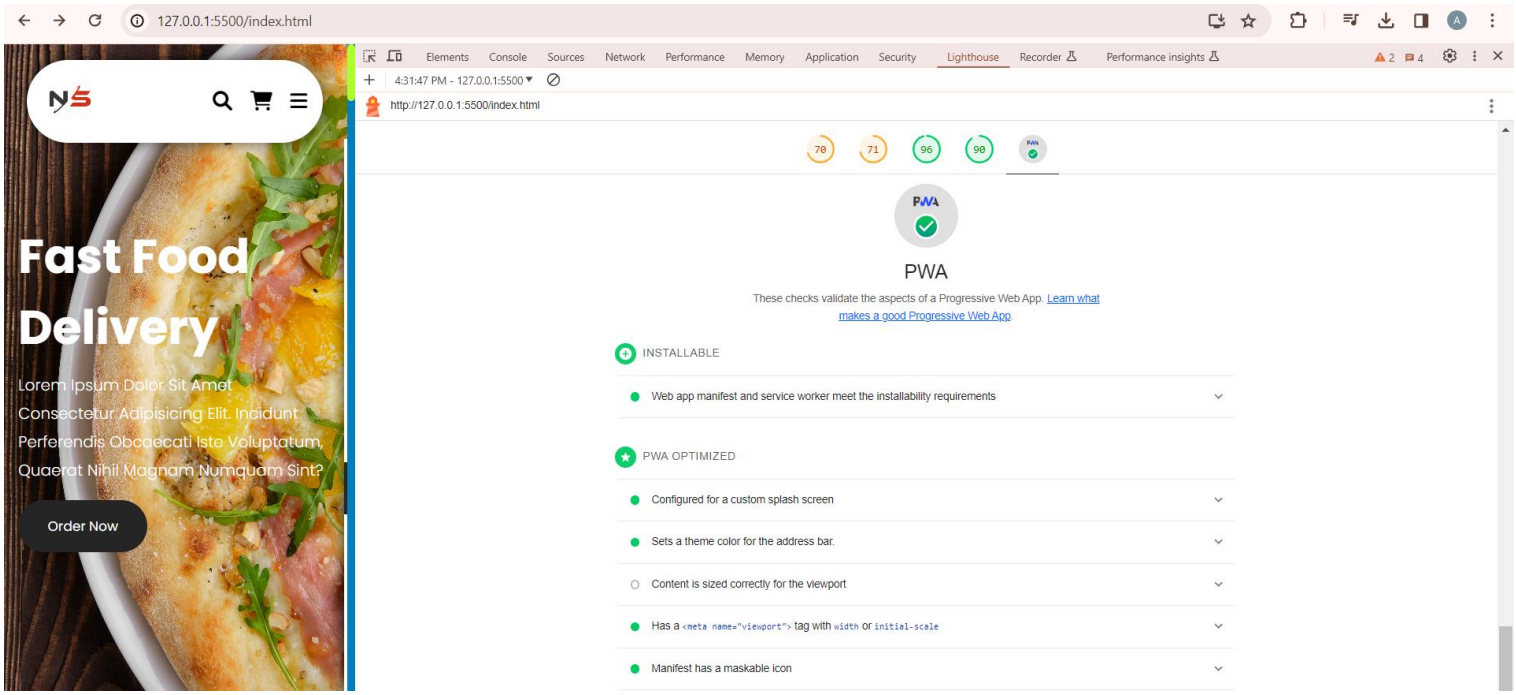
For theme color add a meta tag in index.html-

<meta name="theme-color" content="#4285f4">

For a maskable icon add "purpose": "any maskable" to the icons in manifest.json file

For apple touch icon add the following meta tag in index.html-

<link rel="apple-touch-icon" href="">



Conclusion:

- We analyzed the website with the help of lighthouse tool and found some issues with the website
- Hence we made changes in the manifest.json file we added "purpose":"any maskable" for maskable error face
- Also added a meta tag in index.html to resolve the theme error faced

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.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO1: Understand cross platform mobile application development using Flutter framework LO2: Design and Develop interactive Flutter App by using widgets, layouts, gestures and animation LO3: Analyze and Build production ready Flutter App by incorporating backend services and deploying on Android / iOS
Grade:	5

ATHARVA CHAVAN
D15A/10

FLUTTER ASSIGNMENT

Q.1] Key features of Flutter :

1. Single codebase for multiple platforms - Flutter allows developers to write code & deploy it on both iOS & Android platform.
2. Hot Reload - This enables developers to instantly see the results of the code changes they make.
3. Expressive UI - Developers have the flexibility to create expressive & flexible UI.
4. Integration with other tools - Flutter can easily integrate with other popular development tools & frameworks.

Advantages of Flutter :

1. Faster development - Uses single codebase for multiple platforms.
2. Consistent UI across platforms - Widgets provide a consistent look & feel across different platforms.
3. Cost efficiency - Developing & maintaining single codebase for both iOS & Android reduces development cost & resources.

Differs from Traditional Approach :

1. Traditional approach uses a hierarchial structure for UI components whereas Flutter uses widget - based approach.
2. Flutter compiles to native ARM code, providing performance comparable to native applications.
3. Hot reloads allow to see changes made instantly.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

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

Q. 2] Widget Tree :

① The widget tree is hierarchial structure of widgets that define the user interface of an application.

② Every visual element from simple components to complex layout is represented by a widget.

③ Widget can be categorized in 2 types :

A] Stateless Widget :
It is immutable & cannot change over time.
eg :- text, images, etc.

B] Stateful Widget :
Widget that can change its state over time
eg :- buttons, form

Widget Composition :

① Widget composition in Flutter involves combining multiple simple widget to create more complex & compound widget.

② This composability is powerful concept that allows developers to build sophisticated user interfaces by nesting widgets within each other.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

Commonly Used Widget :

(1) Container - A box model for padding, margin & decoration.
(2) Column & Row - Layout widgets for arranging children vertically or horizontally.
(3) Stack - Overlapping widgets, allowing them to be layered on top of each other.
(4) List view - A scrollable list of widget
(5) Grid view - A scrollable grid of widgets
(6) AppBar - A material design app bar typically at the top of screen.
(7) Text Field - An input field for users to enter text
(8) Button widget - Interactive buttons for user actions.

Q. 3] State Management in Flutter :

(1) State management is crucial in flutter application because it involves managing the data that can change over time.
(2) Flutter is reactive, meaning the UI rebuilds when the underlying data changes.

	set State	Provider	Riverpod
(1)	Built in flutter method.	(1) External package named ('provider')	(1) External package ('riverpod')

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

(2)	Local state within a widget.	(2) Global state within a widget tree.	(2) Global state with additional features.
(3)	Limited scalability for large apps.	(3) Suitable for medium sized apps.	(3) Design for large & complex apps.
(4)	May lead to code redundancy.	(4) Balances simplicity & readability	(4) Emphasizes readability & clear syntax.
(5)	Testing can be more challenging	(5) Good testability support.	(5) Enhanced testing experience.
(6)	Widely used & well established.	(6) Widely adopted & well supported	(6) Gaining popularity & growing community.

Scenario where each is applicable :

(1) set State :

i) For small to moderately complex application

ii) When managing local state within a widget

eg :- simple forms, UI components with local UI specific state.

Sundaram

FOR EDUCATIONAL USE

Scanned with OKEN Scanner

(2) Provider :

i) For medium to large scale application.

ii) When a centralized state is used within a widget.

eg :- Managing user authentication, theme change or app-wide configuration.

(3) Riverpod :

i) For large & complex application.

ii) When testability & maintainability are top priorities.

eg :- complex application with multiple feature, dynamic UI.

Q.4] Integration :

(1) Go to Firebase console & create a new project.

(2) Add Firebase SDK by adding dependencies in pubspec.yaml file.

dependencies :

firebase_core : ^ version

firebase_auth : ^ version

cloud_firebase : ^ version

(3) Run flutter get pub

(4) Initialise Firebase by calling :
'Firebase.initialiseApp()' in main

Sundaram®

FOR EDUCATIONAL USE

Scanned with OKEN Scanner

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

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

- Benefits of using Firebase as Backend :

(1) Real time database :-
Firebase offers real time NoSQL database

(2) Authentication :-
Provides a secure & easy-to implement solution for user authentication.

(3) Cloud firebase :-
Firebase's clould firestore provides a secure & easy-to implement scalable NoSQL database that allows you to store & sync data in real time.

(4) Hosting :
Firebase hosting provides a simple & efficient way to deploy & host web application.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

Data Synchronization :

(1) Real-time database :-
When data changes on one client, it triggers event automatically update data on other clients.

(2) Cloud Firestore :-
It notifies client when data changes, allowing for seamless real-time updates.

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

Sundarani®
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

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

Q. 2] Widget Tree :

① The widget tree is hierarchial structure of widgets that define the user interface of an application.

② Every visual element from simple components to complex layout is represented by a widget.

③ Widget can be categorized in 2 types :

A] Stateless Widget :
It is immutable & cannot change over time.
eg :- text, images, etc.

B] Stateful Widget :
Widget that can change its state over time
eg :- buttons, form

Widget Composition :

① Widget composition in Flutter involves combining multiple simple widget to create more complex & compound widget.

② This composability is powerful concept that allows developers to build sophisticated user interfaces by nesting widgets within each other.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

Commonly Used Widget :

(1) Container - A box model for padding, margin & decoration.
(2) Column & Row - Layout widgets for arranging children vertically or horizontally.
(3) Stack - Overlapping widgets, allowing them to be layered on top of each other.
(4) List view - A scrollable list of widget
(5) Grid view - A scrollable grid of widgets
(6) AppBar - A material design app bar typically at the top of screen.
(7) Text Field - An input field for users to enter text
(8) Button widget - Interactive buttons for user actions.

Q. 3] State Management in Flutter :

(1) State management is crucial in flutter application because it involves managing the data that can change over time.
(2) Flutter is reactive, meaning the UI rebuilds when the underlying data changes.

	set State	Provider	Riverpod
(1)	Built in flutter method.	(1) External package named ('provider')	(1) External package ('riverpod')

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

(2)	Local state within a widget.	(2) Global state within a widget tree.	(2) Global state with additional features.
(3)	Limited scalability for large apps.	(3) Suitable for medium sized apps.	(3) Design for large & complex apps.
(4)	May lead to code redundancy.	(4) Balances simplicity & readability	(4) Emphasizes readability & clear syntax.
(5)	Testing can be more challenging	(5) Good testability support.	(5) Enhanced testing experience.
(6)	Widely used & well established.	(6) Widely adopted & well supported	(6) Gaining popularity & growing community.

Scenario where each is applicable :

(1) set State :

i) For small to moderately complex application

ii) When managing local state within a widget

eg :- simple forms, UI components with local UI specific state.

Sundaram

FOR EDUCATIONAL USE

Scanned with OKEN Scanner

(2) Provider :

i) For medium to large scale application.

ii) When a centralized state is used within a widget.

eg :- Managing user authentication, theme change or app-wide configuration.

(3) Riverpod :

i) For large & complex application.

ii) When testability & maintainability are top priorities.

eg :- complex application with multiple feature, dynamic UI.

Q.4] Integration :

(1) Go to Firebase console & create a new project.

(2) Add Firebase SDK by adding dependencies in pubspec.yaml file.

dependencies :

firebase_core : ^ version

firebase_auth : ^ version

cloud_firebase : ^ version

(3) Run flutter get pub

(4) Initialise Firebase by calling :
'Firebase.initialiseApp()' in main

Sundaram®

FOR EDUCATIONAL USE

Scanned with OKEN Scanner

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

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

- Benefits of using Firebase as Backend :

(1) Real time database :-
Firebase offers real time NoSQL database

(2) Authentication :-
Provides a secure & easy-to implement solution for user authentication.

(3) Cloud firebase :-
Firebase's clould firestore provides a secure & easy-to implement scalable NoSQL database that allows you to store & sync data in real time.

(4) Hosting :
Firebase hosting provides a simple & efficient way to deploy & host web application.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

Data Synchronization :

(1) Real-time database :-
When data changes on one client, it triggers event automatically update data on other clients.

(2) Cloud Firestore :-
It notifies client when data changes, allowing for seamless real-time updates.

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

Sundarani®
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

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.	10
Name	Atharva Chavan
Class	D15A
Subject	MAD & PWA Lab
Lab Outcome	LO4:Understand various PWA frameworks and their requirements LO5: Design and Develop a responsive User Interface by applying PWA Design techniques LO6:Develop and Analyze PWA Features and deploy it over app hosting solutions
Grade:	4

ATHARVA CHAVAN
D15A/10

PWA ASSIGNMENT

Q.1]

(1) Progressive Web Apps (PWA) are type of web applications that combines the best features of traditional web applications & native Mobile applications.

(2) PWA are built using web technologies like HTML, CSS & JS but they can provide a user experience that is similar to a native code.

(3) Significance in modern app development

3) PWA are significant in modern web development because they offer a number of advantages over a traditional app & native mobile app like:

i) Reachable: They can be accessed from any device with a web browser, regardless of OS.

ii) Reliable: They can work offline, thanks to service-workers.

iii) Instantable: They can be installed on user's home screen just like native app.

iv) Engaging: They can provide a user experience that is indistinguishable from native app.

(4) Key characteristics that differentiate from traditional web apps:

i) PWA's are built using web technologies, while native mobile apps are built using platform specific programming languages.

eg:- Swift for iOS, Java for Android

ii) PWA's do not require installation from an app store, while native mobile apps typically do.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

iii) PWA's can work offline, while traditional mobile apps typically can't.

Q. 2] (1) Responsive Web Design (RWD): It is a web design approach that ensures that a website looks good & functions properly on all devices. from desktop computers to tablets & smartphones.

(2) RWD is important for PWA because it ensures that the app can be used on variety of devices.

(3) Comparison of Responsive, fluid & adaptive web design approaches:

i) Responsive web design :- A RWD uses a single flexible layout that adapts to different screen size. This is the most common approach for PWA.

ii) Fluid web design :- A fluid web design uses a flexible layout that scales to fit the width of browser window. This can be good option for websites that have lot of content that needs to be resized to fit different screens.

iii) Adaptive web designs :- An adaptive web design uses a different layout for different screen sizes. This can be good option for websites that need to provide different user experience for different devices.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner

Q.3] (1) Service-workers :- Service workers are the scripts that run in the background of web application. They can be used to enable features such as push notification, offline functionality & background synchronization.

(2) Lifecycle of Service-workers :-

i) Registration : The service worker script is registered with the browser.

ii) Installation : The service-worker script is downloaded & cached by the browser.

iii) Activation : The service worker takes control of web page or group of web pages.

Q.4] IndexedDB is a browser API for storing large amount of structured data on client-side. It functions like a NoSQL database, allowing you to store key-value pairs & organise data in object stores with indexes for efficient retrieval.

Benefits of using IndexedDB with service worker :

(1) Offline data access : Improves user experience by allowing interaction with PWA even without an internet connection.

(2) Faster load time : Cached data retrieval from IndexedDB can be served much faster than

Sundaram
FOR EDUCATIONAL USE

Scanned with OKEN Scanner

fetching it from server again.

(3) Improved reliability : Reduces dependence on constant N/W connection making the PWA more reliable.

(4) Security : Indexed DB provides a server storage mechanism within the browser's sandbox environment

(5) Data Management : PWA should have mechanisms to manage storage space & avoid excessive data accumulation in Indexed DB.

Sundaram
FOR EDUCATIONAL USE
Scanned with OKEN Scanner