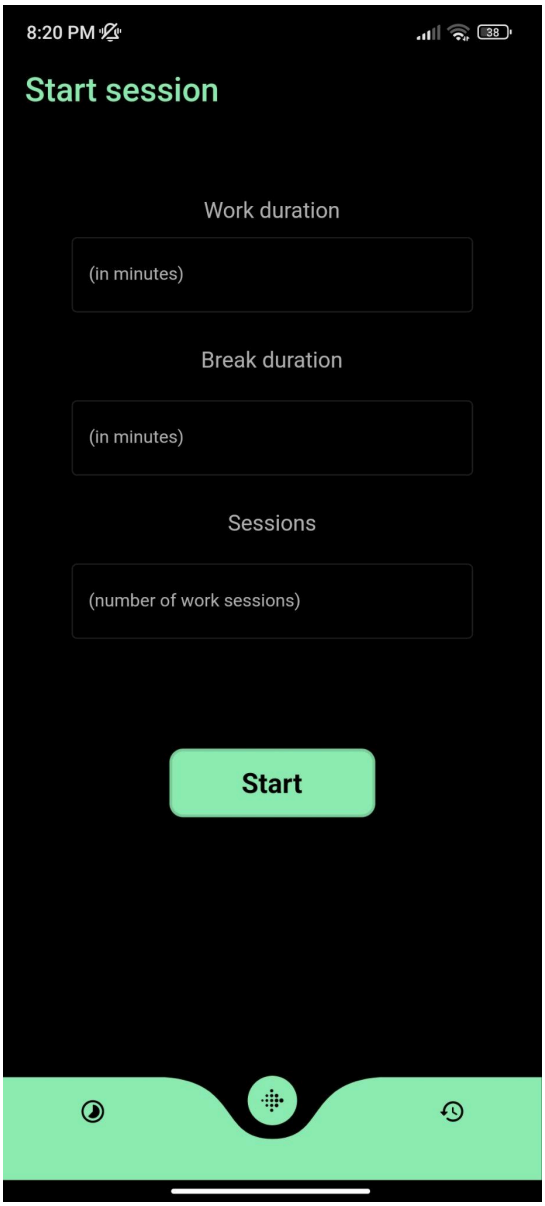
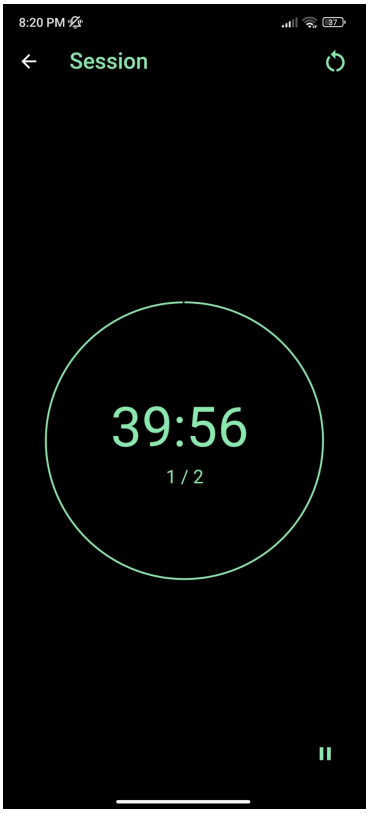
MAD and PWA Lab

Name: Tanmay Bhosale **Class:** D15A **Roll no: 8**

Aim: Selecting features for application development, the features should comprise of:

1. Common widgets
2. Should include icons, images, charts etc.
3. Should have an interactive Form
4. Should apply navigation, routing and gestures
5. Should connect with FireBase database

Screen-shot	Features
1. 	**Home-screen** 2. Multiple input widgets making it a form page 3. Navigation using the buttons at the Bottom and the top. 4. Icons for the buttons in the bottom navigation bar. 5. Start button to start the session and move on to the next page.

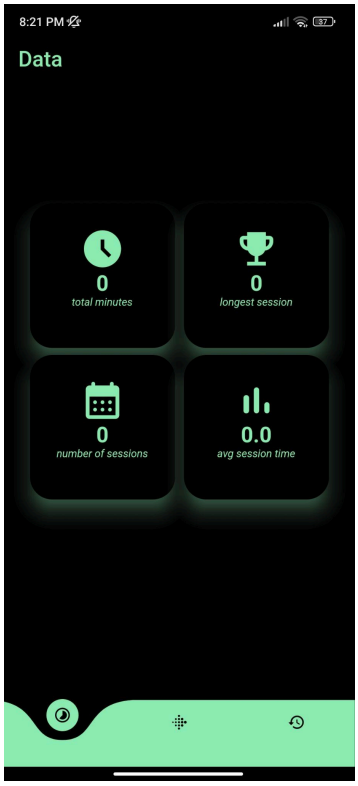


Main session page:

1. Timer Widget with countdown
2. Button to pause the session at the bottom.
3. Button to restart the session at the top.

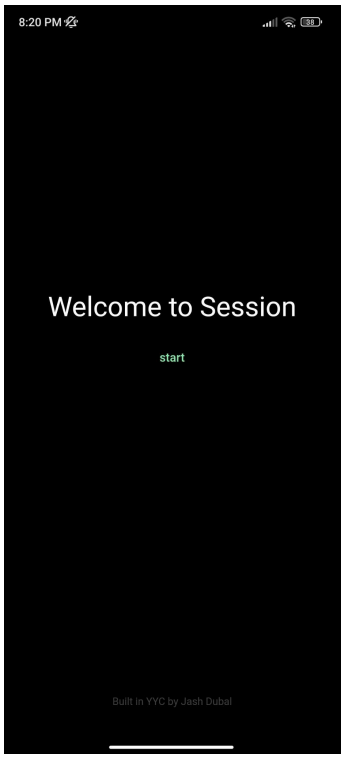
History Page

1. History of the previous sessions is shown using small cards.
2. Button to delete a specific history in the top-right corner of the page.



Statistics Page:

1. Shows the statistics from all the previous sessions. Does this with the help of cards widget:
    a. Total minutes
    b. Longest session
    c. Number of sessions
    d. Average session time



Welcome Page:

1. Welcome page at the start of the app.
2. ‘start’ button to start a session and move to the next page.

Experiment 1

Tanmay Bhosale
D15A 08

Aim: To install and configure the Flutter Environment

Theory:

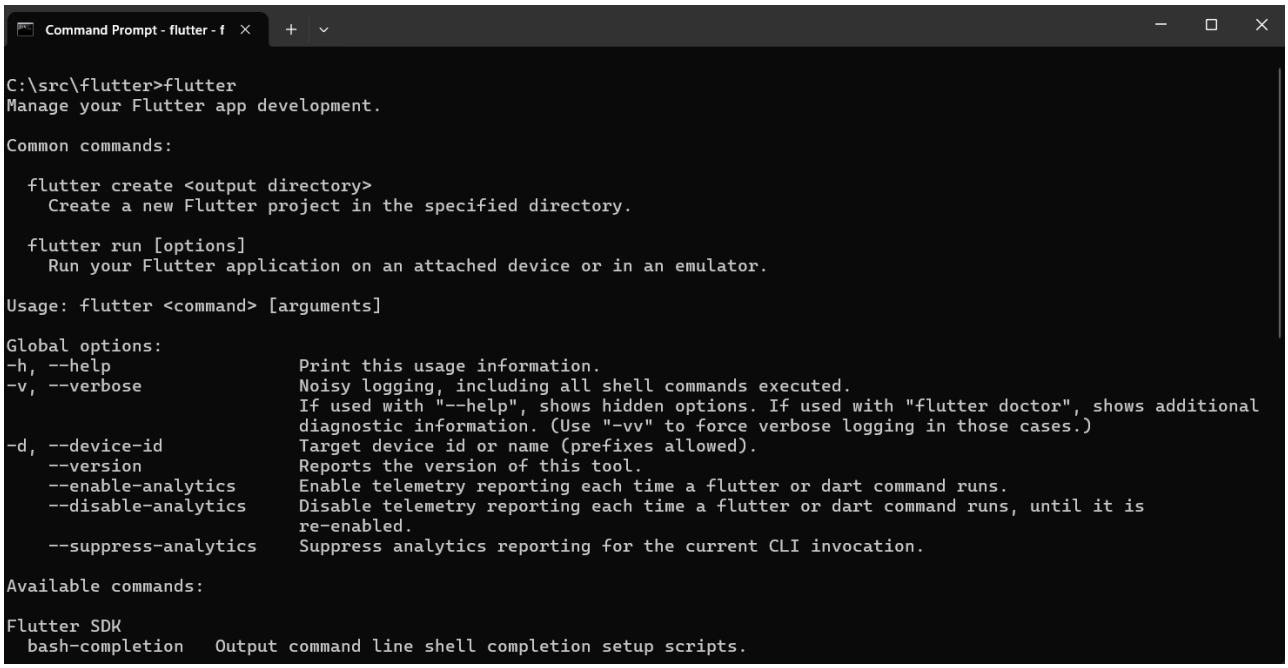
Flutter is an open-source UI software development toolkit created by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. It was first introduced in 2015 and has gained popularity for its ability to create high-performance and visually appealing applications with a smooth and consistent user experience.

Key features of Flutter include:

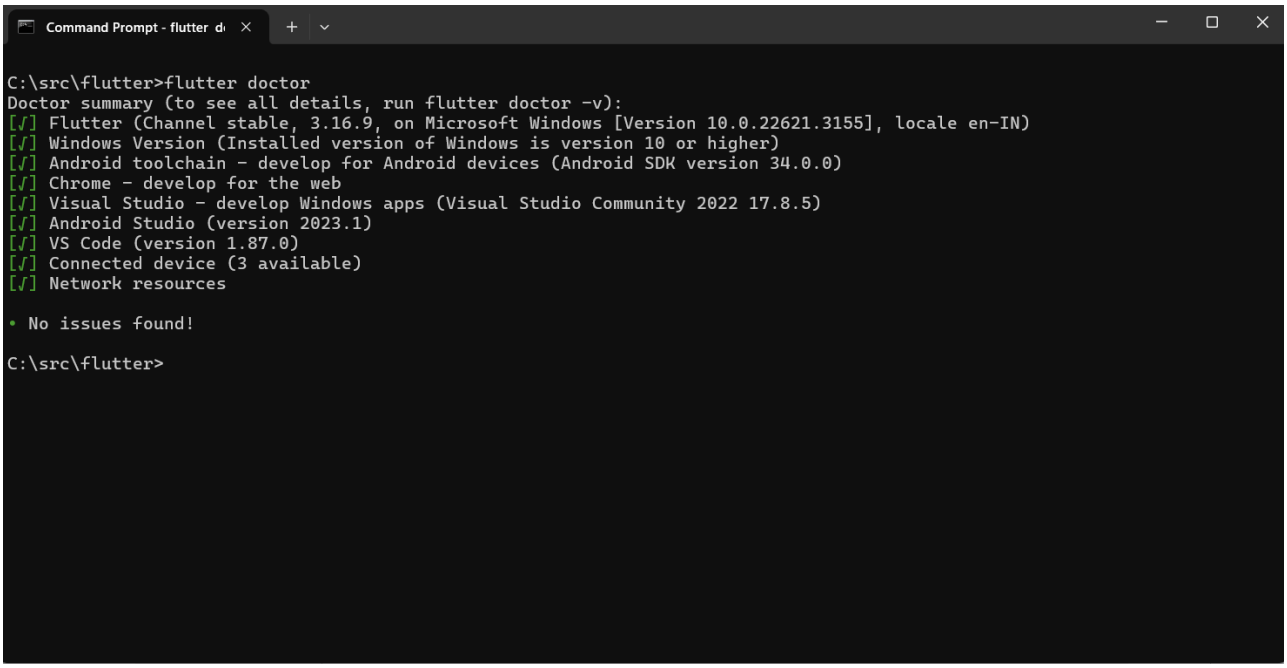
1. **Single Codebase:** With Flutter, you can write your application code once and deploy it on multiple platforms, such as iOS, Android, web, and desktop. This reduces development time and effort, as you don't need separate codebases for each platform.
2. **Widgets:** Flutter uses a reactive framework with a wide range of customizable widgets that allow developers to create complex and interactive user interfaces. Widgets are building blocks for the UI, and Flutter provides a rich set of both material design (for Android) and Cupertino (for iOS) widgets.
3. **Hot Reload:** One of Flutter's standout features is Hot Reload, which enables developers to instantly see the results of code changes in their running application. This significantly speeds up the development and debugging process.
4. **High Performance:** Flutter compiles to native ARM code for mobile devices, providing high performance comparable to native applications. It also has a GPU-accelerated rendering engine called Skia, contributing to smooth animations and a responsive user interface.
5. **Dart Programming Language:** Flutter applications are typically written in Dart, a modern, object-oriented language developed by Google. Dart is known for its simplicity, efficiency, and ease of learning.
6. **Community and Ecosystem:** Flutter has a growing and active community of developers and a rich ecosystem of packages and plugins available through the Flutter pub.dev repository. This allows developers to easily integrate various functionalities into their applications.

- Android Studio is the official integrated development environment (IDE) for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.

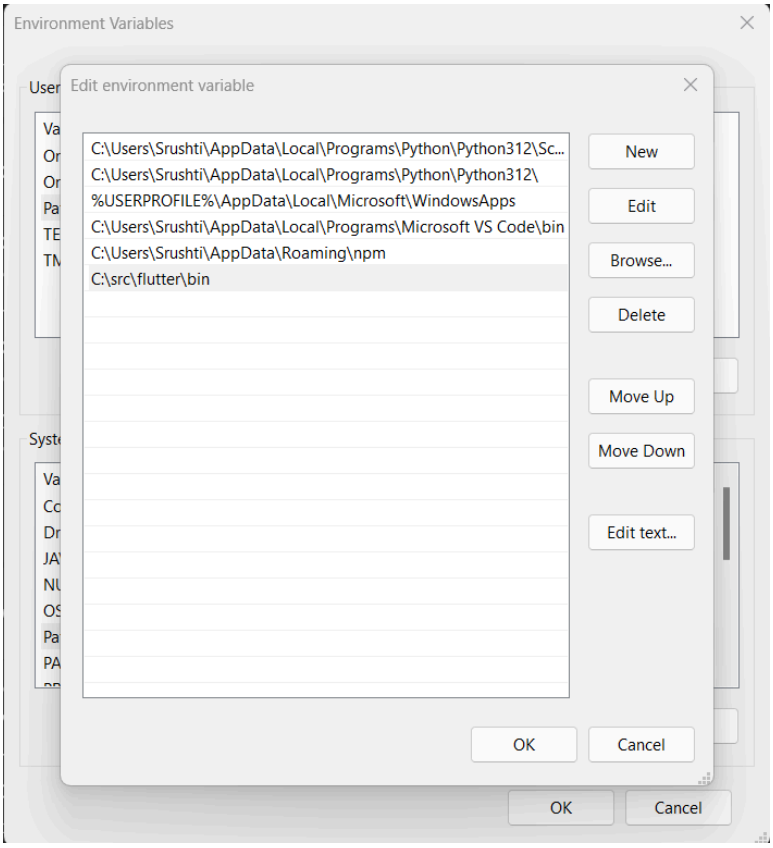
Flutter is successfully installed



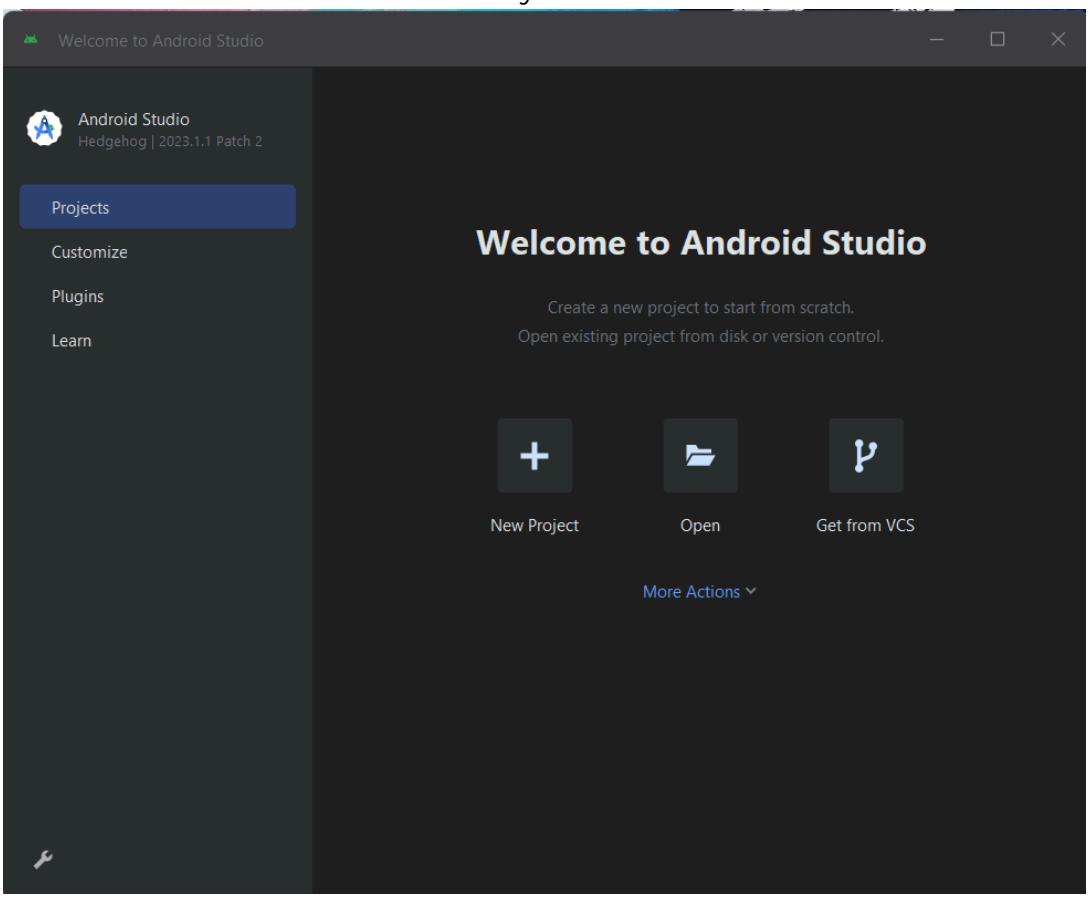
Flutter Doctor:



Flutter and related dependencies are installed correctly and added to the path environment variables.



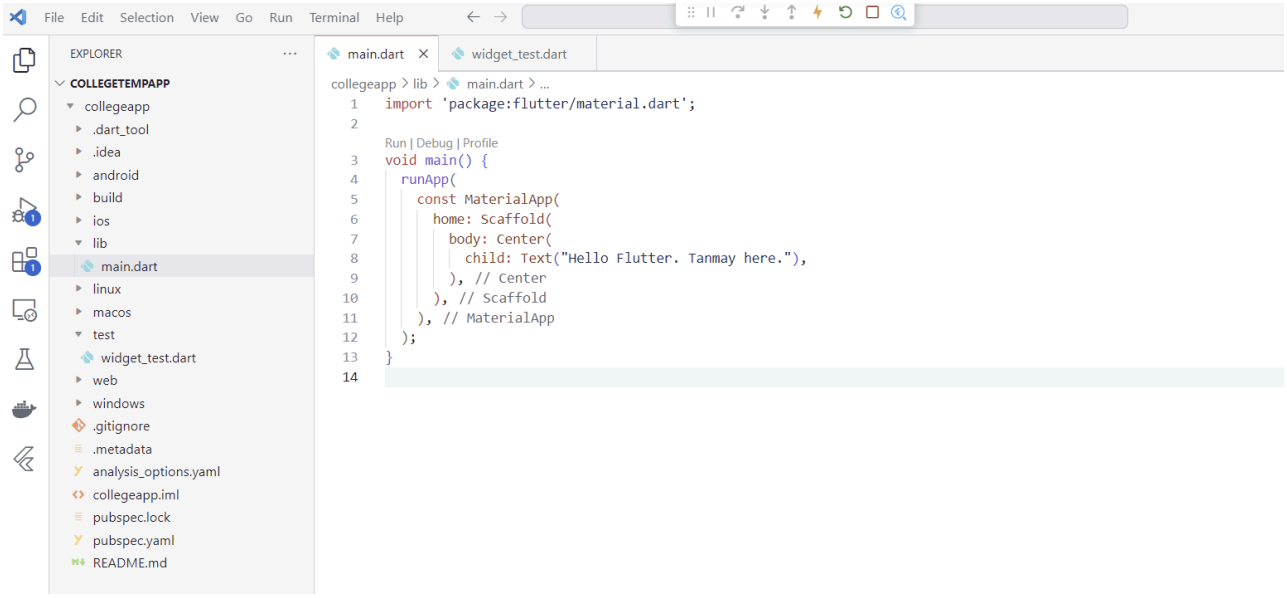
Installed Android Studio successfully:



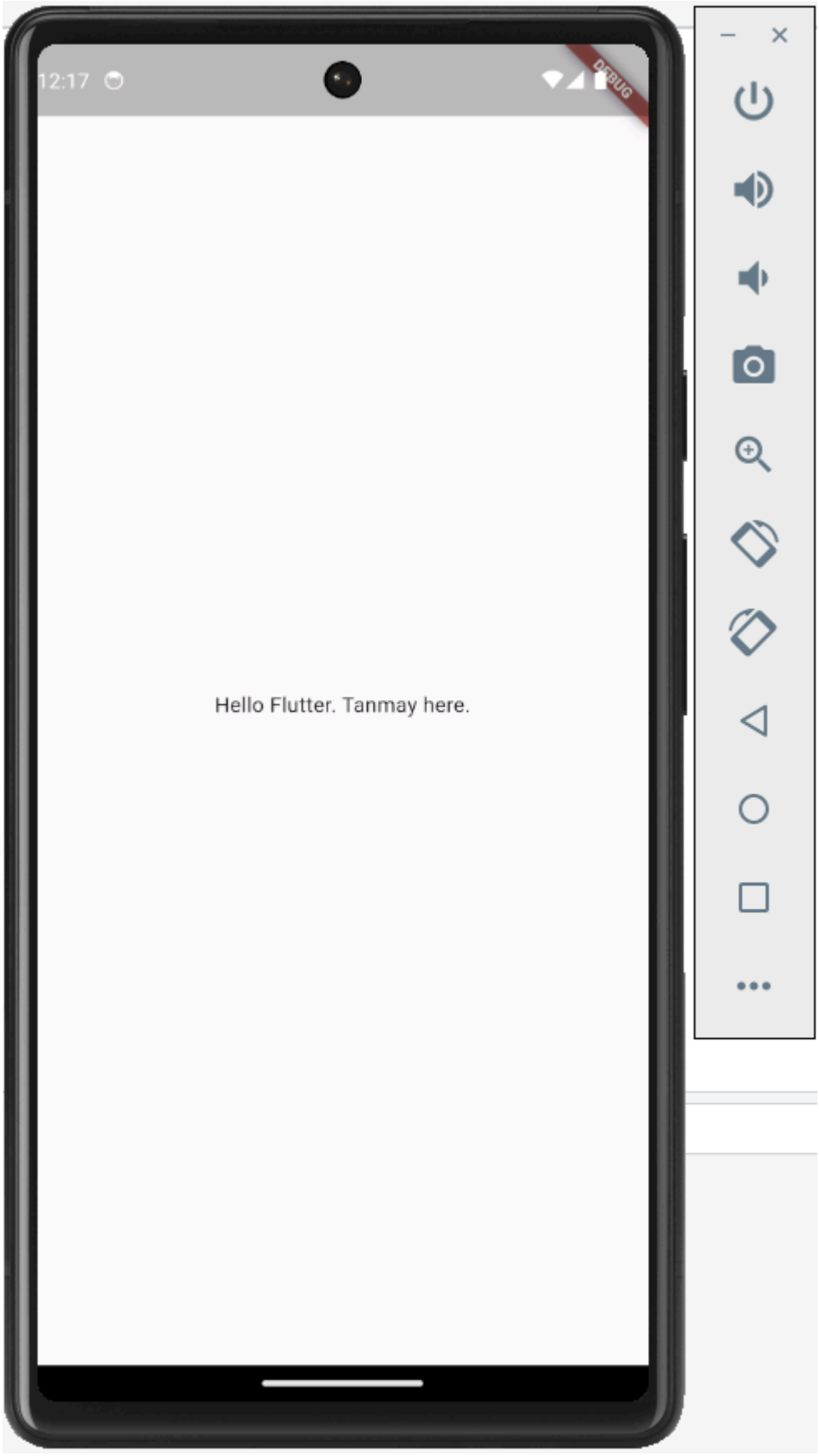
Entering code into the main.dart file:

Code:

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

void main() {
  runApp(
    const MaterialApp(
      home: Scaffold(
        body: Center(
          child: Text("Hello Flutter. Tanmay here."),
        ),
      ),
    ),
  );
}
```



Output:



Experiment 2

Tanmay Bhosale
D15A 08

Aim: To design Flutter UI by including common widgets.

Theory:

Widgets: Each element on a screen of the Flutter app is a widget. The view of the screen completely depends upon the choice and sequence of the widgets used to build the apps. And the structure of the code of an apps is a tree of widgets.

Category of Widgets:

There are mainly 14 categories in which the flutter widgets are divided. They are mainly segregated on the basis of the functionality they provide in a flutter application.

1. Accessibility: These are the set of widgets that make a flutter app more easily accessible.
2. Animation and Motion: These widgets add animation to other widgets.
3. Assets, Images, and Icons: These widgets take charge of assets such as display images and show icons.
4. Async: These provide async functionality in the flutter application.
5. Basics: These are the bundle of widgets that are absolutely necessary for the development of any flutter application.
6. Cupertino: These are the iOS designed widgets.
7. Input: This set of widgets provides input functionality in a flutter application.
8. Interaction Models: These widgets are here to manage touch events and route users to different views in the application.
9. Layout: This bundle of widgets helps in placing the other widgets on the screen as needed.
10. Material Components: This is a set of widgets that mainly follow material design by Google.
11. Painting and effects: This is the set of widgets that apply visual changes to their child widgets without changing their layout or shape.
12. Scrolling: This provides scrollability of to a set of other widgets that are not scrollable by default.
13. Styling: This deals with the theme, responsiveness, and sizing of the app.
14. Text: This displays text.

Description of few of the widgets are as follows:

- Scaffold – Implements the basic material design visual layout structure.
- App-Bar – To create a bar at the top of the screen.
- Text - To write anything on the screen.
- Container – To contain any widget.
- Center – To provide center alignment to other widgets.

The code in main.dart:

```
import 'package:flutter/material.dart';
import 'package:hp/landing.dart';
import 'package:window_manager/window_manager.dart';
import 'dart:io';


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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Builder(
        builder: (BuildContext context) {
          final mediaQueryData = MediaQuery.of(context);
          const screenWidth = 250.0;
          const screenHeight = 400.0;
          return MediaQuery(
            data: mediaQueryData.copyWith(
              size: const Size(screenWidth, screenHeight),
              devicePixelRatio: mediaQueryData.devicePixelRatio,
            ),
            child: const SizedBox(
              width: screenWidth,
              height: screenHeight,
              child: LandingPage(),
            ),
          );
        },
      ),
    );
  }
}
```

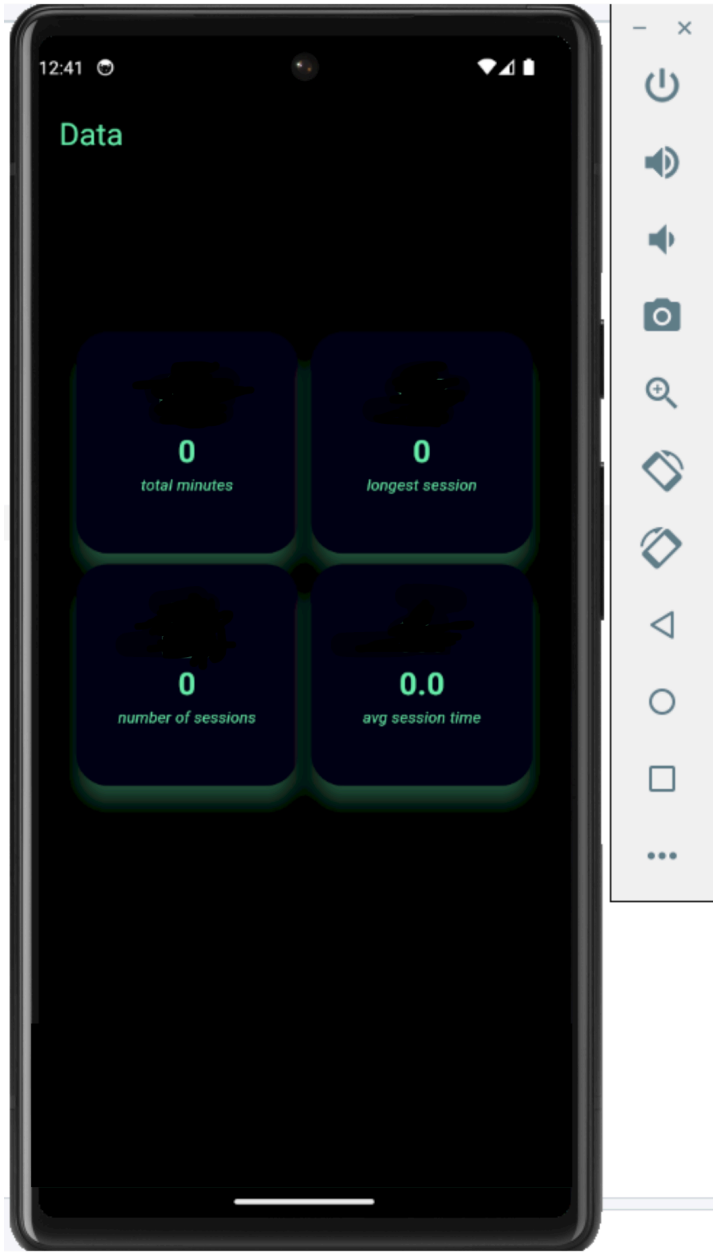
The code in data.dart:

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

class Data extends StatefulWidget {
  const Data({Key? key}) : super(key: key);

  @override
  State<StatefulWidget> createState() {
    return _DataState();
  }
}
@override
  Widget build(BuildContext context) {
    return Scaffold(
        backgroundColor: Colors.black,
        // padding: const EdgeInsets.all(20.0),
        appBar: AppBar(
            automaticallyImplyLeading: false,
            centerTitle: false,
            backgroundColor: Colors.black,
            title: const Text.rich(
              TextSpan(
                text: 'Data', // text for title
                style: TextStyle(
                  fontSize: 24,
                  color: Colors.greenAccent,
                  fontFamily: 'Arial',
                ),
              ),
            )),
        body: SizedBox(
            width: 600,
            height: 600,
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Padding(
                  padding: const EdgeInsets.all(25.0),
```

```
child: GridView(
  shrinkWrap: true,
  gridDelegate:
      const SliverGridDelegateWithFixedCrossAxisCount(
    crossAxisCount: 2,
    crossAxisSpacing: 2,
    mainAxisSpacing: 0,
  ),
  children: [
    Card(
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(25.0),
      ),
      clipBehavior: Clip.antiAlias,
      color: Colors.black,
      shadowColor: Colors.greenAccent,
      elevation: 15,
      child: Padding(
        padding: const EdgeInsets.all(26.0),
        child: Column(
          children: [
            const Icon(
              Icons.access_time_filled_outlined,
              color: Colors.greenAccent,
              size: 50,
            ),
            Text(_totalMin.toString(),
                style: const TextStyle(
                    color: Colors.greenAccent,
                    fontSize: 24,
                    fontWeight: FontWeight.bold,
                    fontFamily: 'Arial')),
            const Text('total minutes',
                style: TextStyle(
                    color: Colors.greenAccent,
                    fontSize: 12,
                    fontFamily: 'Arial',
                    fontStyle: FontStyle.italic)),
          ],
```

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

Output:


12:41
Data
0
total minutes
0
longest session
0
number of sessions
0.0
avg session time

Experiment 3

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

Theory:

A flutter app when built has both assets (resources) and code. Assets are available and deployed during runtime. The asset is a file that can include static data, configuration files, icons, and images. The Flutter app supports many image formats, such as JPEG, WebP, PNG, GIF, animated WebP/GIF, BMP, and WBMP.

Steps to Add an Image:

Step 1. Create a new folder

- It should be in the root of your flutter project. You can name it whatever you want, but assets are preferred.
- If you want to add other assets to your app, like fonts, it is preferred to make another subfolder named images.

Step 2. Now you can copy your image to images sub-folder. The path should look like assets/images/yourImage. Before adding images also check the above-mentioned supported image formats.

Step 3. Register the assets folder in pubspec.yaml file and update it. To add images, write the following code:

```
flutter:
  assets:
      - assets/images/yourFirstImage.jpg
      - assets/image/yourSecondImage.jpg
```

If you want to include all the images of the assets folder then add this:

```
flutter:
  assets:
      - assets/images/
```

Step 4. Insert the image code in the file, where you want to add the image.

Step 5. Now you can save all the files and run the app, you will find the output as shown below.

Icon class in Flutter is used to show specific icons in our app. Instead of creating an image for our icon, we can simply use the Icon class for inserting an icon in our app. For using this class you must ensure that you have set uses-material-design: true in the pubsec.yml file of your object.

Properties:

- color: It is used to set the color of the icon
- size: It is used to resize the Icon
- semanticLable: It comes in play while using the app in accessibility mode (ie, voice-over)
- textDirection: It is used for rendering

Similarly Icons can be added to the app.

Code in pubspec.yaml for adding icons:

```
# The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.2
  curved_navigation_bar: ^1.0.3
  async: ^2.9.0
  animated_snack_bar: ^0.3.0
  shared_preferences: ^2.0.15
  intl: ^0.18.0
  badges: ^2.0.3
  rename: ^2.0.1
  window_manager: ^0.0.1
```

Code in data.dart for adding icons:

```
child: Column(
                              children: [
                                const Icon(
                                  Icons.access_time_filled_outlined,
                                  color: Colors.greenAccent,
                                  size: 50,
                                )
```

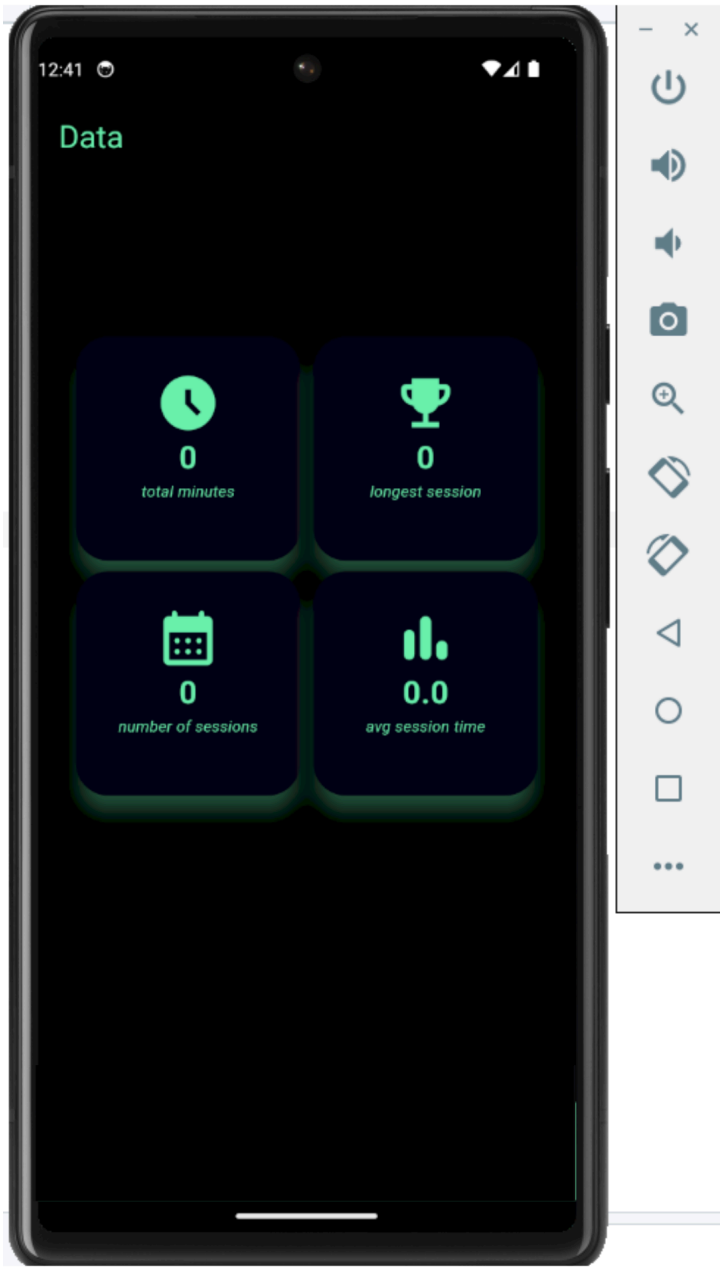
Code for the card displaying the icon:

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
        backgroundColor: Colors.black,
        // padding: const EdgeInsets.all(20.0),
        appBar: AppBar(
            automaticallyImplyLeading: false,
            centerTitle: false,
            backgroundColor: Colors.black,
            title: const Text.rich(
              TextSpan(
```

```
          text: 'Data', // text for title
          style: TextStyle(
            fontSize: 24,
            color: Colors.greenAccent,
            fontFamily: 'Arial',
          ),
        ),
      )),
  body: SizedBox(
      width: 600,
      height: 600,
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Padding(
            padding: const EdgeInsets.all(25.0),
            child: GridView(
              shrinkWrap: true,
              gridDelegate:
                  const SliverGridDelegateWithFixedCrossAxisCount(
                crossAxisCount: 2,
                crossAxisSpacing: 2,
                mainAxisSpacing: 0,
              ),
              children: [
                Card(
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(25.0),
                  ),
                  clipBehavior: Clip.antiAlias,
                  color: Colors.black,
                  shadowColor: Colors.greenAccent,
                  elevation: 15,
                  child: Padding(
                    padding: const EdgeInsets.all(26.0),
                    child: Column(
                      children: [
                        const Icon(
                          Icons.access_time_filled_outlined,
                          color: Colors.greenAccent,
                          size: 50,
                        ),
```

```
            Text(_totalMin.toString(),
                style: const TextStyle(
                    color: Colors.greenAccent,
                    fontSize: 24,
                    fontWeight: FontWeight.bold,
                    fontFamily: 'Arial')),
            const Text('total minutes',
                style: TextStyle(
                    color: Colors.greenAccent,
                    fontSize: 12,
                    fontFamily: 'Arial',
                    fontStyle: FontStyle.italic)),
          ],
        ),
      ),
    ),
```

Output:


12:41
Data
0
total minutes
0
longest session
0
number of sessions
0.0
avg session time

Experiment 4

Tanmay Bhosale
D15A 08

Aim: To create an interactive Form using form widget

Theory:
The Form widget in Flutter is a fundamental widget for building forms. It provides a way to group multiple form fields together, perform validation on those fields, and manage their state.

State Management: Flutter provides various ways to manage the state of interactive forms. You can use StatefulWidget or state management solutions like Provider, Riverpod, or Bloc pattern to handle form data changes efficiently. The choice depends on the complexity and requirements of your application.

Form Widgets: Flutter offers a range of form widgets such as TextField, Checkbox, Radio, DropdownButton, etc., which are used to collect input from users. These widgets can be customized with various parameters to fit the design and functionality of your interactive form.

Validation: Validating user input is essential for ensuring data integrity. Flutter provides a built-in mechanism for form validation using the Form widget along with TextFormField widgets. You can define validation logic for each form field and display error messages accordingly.

Input Handling: Flutter offers various options for handling user input, including keyboard input, gestures, and voice input. You can use event listeners like onChanged, onSubmitted, and onTap to capture user input and update the form state accordingly.

Theming and Styling: Flutter's flexible styling and theming system allow you to customize the appearance of form elements to match your app's design language. You can use themes, colors, fonts, and custom widgets to create visually appealing and consistent interactive forms.

Some Properties of Form Widget

- key: A GlobalKey that uniquely identifies the Form. You can use this key to interact with the form, such as validating, resetting, or saving its state.
- child: The child widget that contains the form fields. Typically, this is a Column, ListView, or another widget that allows you to arrange the form fields vertically.
- autovalidateMode: An enum that specifies when the form should automatically validate its fields.

- **Building a form for taking the information about the session to be set.**
- Information to be entered by the user:
    - Work Duration
    - Break Duration
    - Number of sessions

Code in habit.dart for taking the three required fields:

```
return GestureDetector(
        onTap: () => FocusManager.instance.primaryFocus?.unfocus(),
        child: Scaffold(
            resizeToAvoidBottomInset: false,
            backgroundColor: Colors.black,
            appBar: AppBar(
                automaticallyImplyLeading: false,
                centerTitle: false,
                backgroundColor: Colors.black,
                title: const Text.rich(
                  TextSpan(
                    text: 'Start session', // text for title
                    style: TextStyle(
                      fontSize: 24,
                      color: Colors.greenAccent,
                      fontFamily: 'Arial',
                    ),
                  ),
                )),
            body: SingleChildScrollView(
                child: (Container(
              width: double.infinity,
              color: Colors.black38,
              margin: const EdgeInsets.all(30),
              padding: const EdgeInsets.all(20),
              child: Column(
                children: [
                  const Text(
                    "Work duration",
                    style: TextStyle(
                      fontSize: 16,
                      color: Colors.white70,
                      fontFamily: 'Arial',
                    ),
                  ),
                  const SizedBox(
                      height:
                          10), // add a space between the text and the
```

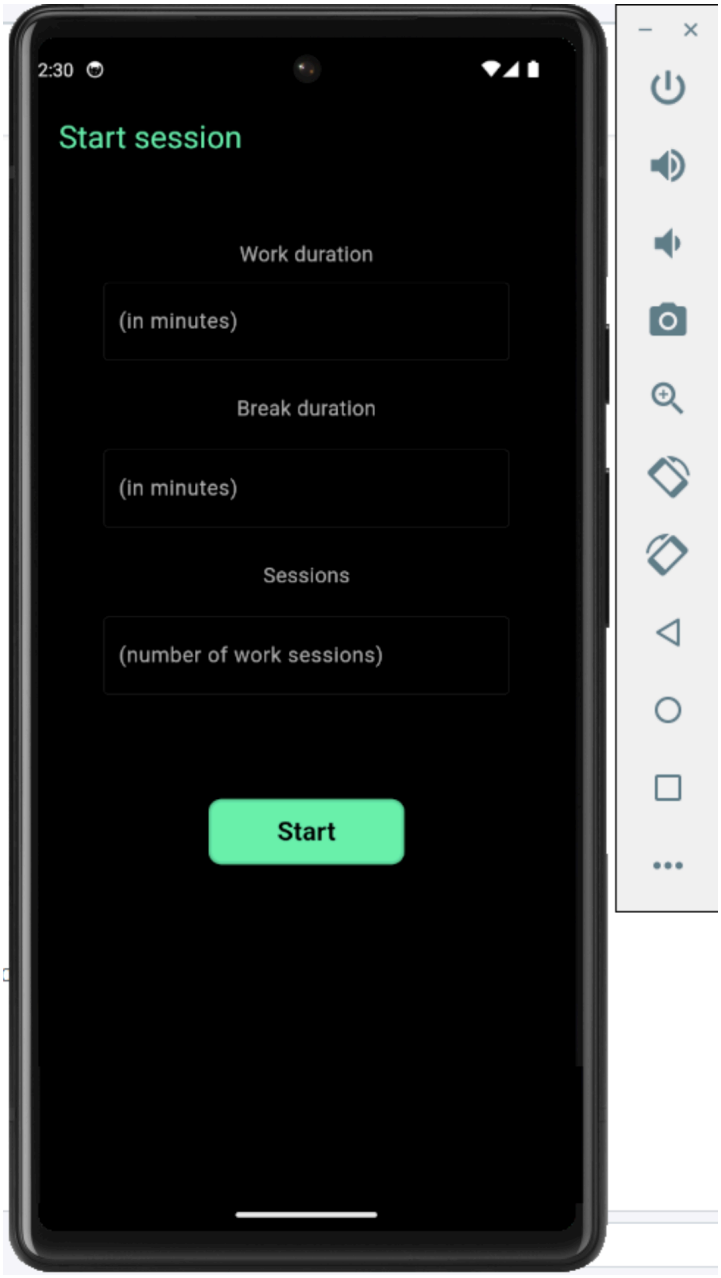
```
input field
                  TextField(
                    controller: workController,
                    textAlign: TextAlign.center,
                    style: const TextStyle(
                      fontSize: 13,
                      color: Colors.white70,
                      fontFamily: 'Arial',
                    ),
                    keyboardType:
                        TextInputType.number, // set the keyboard type to
number
                    keyboardAppearance: Brightness.dark,
                    decoration: const InputDecoration(
                      // filled: true,
                      fillColor: Colors.black12,
                      labelText: '(in minutes)',
                      labelStyle: TextStyle(color: Colors.white70),
                      enabledBorder: OutlineInputBorder(
                          borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                          borderSide: BorderSide(color: Colors.white10)),
                      focusedBorder: OutlineInputBorder(
                          borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                          borderSide: BorderSide(color: Colors.white10)),
                    ),
                  ),
                  const SizedBox(height: 25),
                  const Text(
                    "Break duration",
                    style: TextStyle(
                      fontSize: 16,
                      color: Colors.white70,
                      fontFamily: 'Arial',
                    ),
                  ),
                  const SizedBox(height: 20),
                  TextField(
                    controller: breakController,
                    textAlign: TextAlign.center,
                    style: const TextStyle(
                      fontSize: 13,
                      color: Colors.white70,
```

```
                    fontFamily: 'Arial',
                  ),
                  keyboardType:
                      TextInputType.number, // set the keyboard type to
number
                  keyboardAppearance: Brightness.dark,
                  decoration: const InputDecoration(
                    filled: true,
                    fillColor: Colors.black12,
                    labelText: '(in minutes)',
                    labelStyle: TextStyle(color: Colors.white70),
                    enabledBorder: OutlineInputBorder(
                        borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                        borderSide: BorderSide(color: Colors.white10)),
                    focusedBorder: OutlineInputBorder(
                        borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                        borderSide: BorderSide(color: Colors.white10)),
                  ),
                ),
                const SizedBox(
                    height:
                        25), // add a space between the text and the
input field
                const Text(
                  "Sessions",
                  style: TextStyle(
                    fontSize: 16,
                    color: Colors.white70,
                    fontFamily: 'Arial',
                  ),
                ),
                const SizedBox(height: 20),
                TextField(
                  controller: sessionController,
                  textAlign: TextAlign.center,
                  style: const TextStyle(
                    fontSize: 13,
                    color: Colors.white70,
                    fontFamily: 'Arial',
                  ),
                  keyboardType:
                      TextInputType.number, // set the keyboard type to
```

```
number
                    keyboardAppearance: Brightness.dark,
                    decoration: const InputDecoration(
                      // filled: true,
                      fillColor: Colors.black12,
                      labelText: '(number of work sessions)',
                      labelStyle: TextStyle(
                        color: Colors.white70,
                      ),
                      enabledBorder: OutlineInputBorder(
                          borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                          borderSide: BorderSide(color: Colors.white10)),
                      focusedBorder: OutlineInputBorder(
                          borderRadius:
BorderRadius.all(Radius.circular(4.0)),
                          borderSide: BorderSide(color: Colors.white10)),
                    ),
                  ),
                  const SizedBox(
                      height:
                          80), // add a space between the text and the
input field
                  TextButton(
                    onPressed: () => Navigator.push(
                      context,
                      PageRouteBuilder(
                        transitionDuration: const Duration(seconds: 1),
                        pageBuilder: (context, animation,
secondaryAnimation) {
                          return FadeTransition(
                            opacity: animation,
                            child: MyTimer(
                                breakTime: breakController.text,
                                workTime: workController.text,
                                workSessions: sessionController.text),
                          );
                        },
                      ),
                    ),
                    style: TextButton.styleFrom(
                        backgroundColor: Colors.greenAccent,
                        padding: EdgeInsets.zero,
                        minimumSize: const Size(150, 50),
```

```
                        tapTargetSize: MaterialTapTargetSize.shrinkWrap,
                        alignment: Alignment.center,
                        shape: RoundedRectangleBorder(
                          borderRadius: BorderRadius.circular(10.0),
                          side: const BorderSide(
                              color: Colors.black12, width: 2.0),
                        )),
                    child: const Text(
                      "Start",
                      style: TextStyle(
                        fontSize: 20,
                        color: Colors.black,
                        fontWeight: FontWeight.bold,
                        fontFamily: 'Arial',
                      ),
                    ),
                  ),
                ],
              ),
            ))));
  }
}
```

Output:


2:30
Start session
Work duration
(in minutes)
Break duration
(in minutes)
Sessions
(number of work sessions)
Start

Experiment 5

Tanmay Bhosale
D15A 08

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

Theory:

Flutter offers a comprehensive approach to handle deep links and screen transitions. While the Navigator is suitable for smaller apps with basic deep linking, the Router is recommended for larger apps requiring intricate deep linking. This ensures proper deep link handling across Android, iOS, and web platforms.

Any mobile app's workflow is determined by the user's ability to navigate to various pages; this process is known as routing. The MaterialPageRoute class of Flutter's routing system comes with the two methods that demonstrate how to switch between two routes and they are:

1. Navigator.push() — To navigate to a new route.

2. Navigator.pop() — To return to the previous route.

The Navigator widget manages routes using a stack approach. Routes are stacked based on user actions, and pressing back navigates to the most recent route.

Working of the Navigator:
These following steps show how all the components work together as one piece:
1. The RouteInformationParser converts a new route into a user-defined data type.
2. The RouterDelegate method updates the app state and calls notifyListeners.
3. The Router instructs the RouterDelegate to rebuild via its build() method.
4. The RouterDelegate.build() returns a new Navigator reflecting the updated app state.

Steps to Start Navigation in App:

There are three steps required to start navigation in the app and they are:
1. Create Two Routes: Define the initial and destination routes.
2. Use Navigator.push(): Transition to the new route. To navigate or transition to a new page, route, or screen, the Navigator.push() method is used. The push() method adds a page or route to the stack, and then uses the Navigator to manage it. Once more, we use the MaterialPageRoute class, which enables platform-specific animations for route transitions.
3. Use Navigator.pop(): Navigate back to the initial route. The Navigator controls the stack, and the pop() method enables to remove the current route from it.

The Router and Navigator are designed to work in tandem. By executing imperative methods like push() and pop() on the Navigator, or declarative routing packages like go_router, Router API can ne navigated. Each route on the Navigator is page-backed, which means it was

made from a Page using the pages option on the Navigator constructor, when you navigate using the Router or a declarative routing package. On the other hand, any Route added to the Navigator by calling Navigator.push or showDialog will add a pageless route. Page-backed routes, as opposed to pageless ones, can always be deep-linked when utilising a routing package.

All subsequent pageless routes are also deleted when a page-backed route is deleted from the navigator. For instance, if a deep link navigates by eliminating a page-backed route from the navigator, any pageless _routes that follow up to the following page-backed route are also eliminated.

For a consistent experience when utilising the browser's back and forward buttons, apps that use the Router class integrate with the History API. A History API entry is added to the browser's history stack each time you use the Router to traverse. When the back button is pressed, the Router switches to reverse chronological navigation, which transports the user back to the last

Gestures are used to interact with an application. It is generally used in touch-based devices to physically interact with the application. It can be as simple as a single tap on the screen to a more complex physical interaction like swiping in a specific direction to scrolling down an application. It is heavily used in gaming and more or less every application requires it to function as devices turn more touch-based than ever. In this article, we will discuss them in detail.
Some widely used gestures are mentioned here :

- Tap: Touching the surface of the device with the fingertip for a small duration of time period and finally releasing the fingertip.
- Double Tap: Tapping twice in a short time.
- Drag: Touching the surface of the device with the fingertip and then moving the fingertip in a steadily and finally releasing the fingertip.
- Flick: Similar to dragging, but doing it in a speedier way.
- Pinch: Pinching the surface of the device using two fingers.
- Zoom: Opposite of pinching.
- Panning: Touching the device surface with the fingertip and moving it in the desired direction without releasing the fingertip.

The GestureDetector widget in flutter is used to detect physical interaction with the application on the UI. If a widget is supposed to experience a gesture, it is kept inside the GestureDetector widget. The same widget catches the gesture and returns the appropriate action or response.

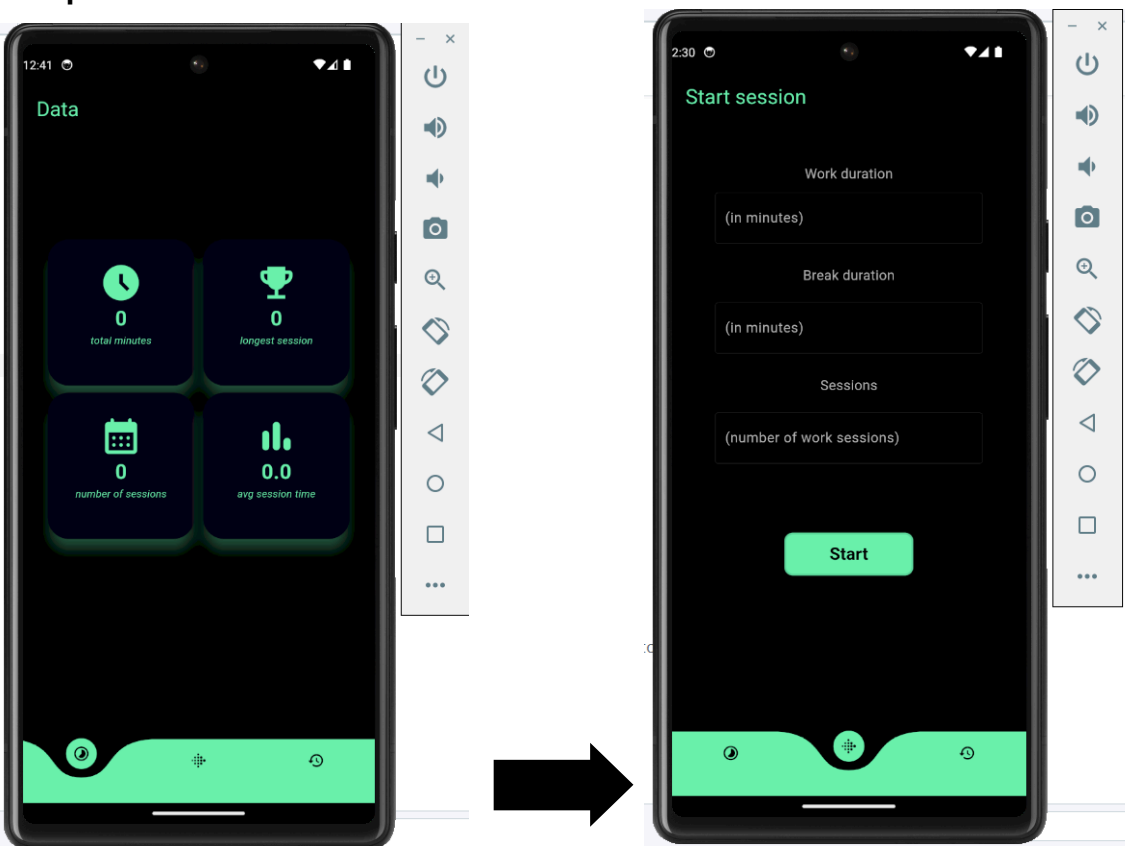
Code in main.dart file:

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

import 'habits.dart'; // Import your page files

import 'data.dart';

import 'settings.dart';


void main() {

  runApp(MyApp());

}


class MyApp extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

        return MaterialApp(

        title: 'Navigation Example',

        initialRoute: '/', // Define the initial route

        routes: {

        '/': (context) => HabotsPage(), // Define route for HabotsPage

        '/data': (context) => DataPage(), // Define route for DataPage

        '/settings': (context) => SettingsPage(), // Define route for SettingsPage

        },

        );

  }

}


class HabotsPage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {
```

```
        return Scaffold(
        appBar: AppBar(
        title: Text('Habits Page'),
        ),
        body: Center(
        child: ElevatedButton(
        onPressed: () {
        // Navigate to the DataPage
        Navigator.pushNamed(context, '/data');
        },
        child: Text('Go to Data Page'),
        ),
        ),
        );
  }
}

class DataPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
        return Scaffold(
        appBar: AppBar(
        title: Text('Data Page'),
        ),
        body: Center(
        child: ElevatedButton(
        onPressed: () {
        // Navigate to the SettingsPage
```

```
        Navigator.pushNamed(context, '/settings');
        },
        child: Text('Go to Settings Page'),
        ),
        ),
        );
  }
}

class SettingsPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
        return Scaffold(
        appBar: AppBar(
        title: Text('Settings Page'),
        ),
        body: Center(
        child: ElevatedButton(
        onPressed: () {
        // Navigate back to the previous screen (DataPage)
        Navigator.pop(context);
        },
        child: Text('Go Back'),
        ),
        ),
        );
  }
}
```

Output:


12:41
Data
0
total minutes
0
longest session
0
number of sessions
0.0
avg session time
2:30
Start session
Work duration
(in minutes)
Break duration
(in minutes)
Sessions
(number of work sessions)
Start

MPL Lab Exp 6

Name: Tanmay Bhosale Roll No. 08

Div: D15A Batch: A

Aim: To Connect Flutter UI with FireBase database

Theory:

Prerequisites

To complete this tutorial, you will need:

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

Code. Create a Firebase Project:

First, log in with your Google account to manage your Firebase projects. From within the Firebase dashboard,

select the Create new project button and give it a name:

Go to the Firebase Console and create a new

project. Add your Flutter app to the Firebase project:

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

The most important thing here is to match up the Android package name that you choose here with the one

inside of our application.

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

name, and the application name:

com.example.flutterfirebaseexample

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

the Android package name:

android/app/build.gradle

...

defaultConfig {

// TODO: Specify your own unique Application ID

(https://developer.android.com/studio/build/application-id.html)

. applicationId 'com.example.flutterfirebaseexample'

...

} ...

Downloading the Config File

The next step is to add the Firebase configuration file into our Flutter project. This is important as it contains

the API keys and other critical information for Firebase to

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

2. Add Firebase to your Flutter project:

Add Dependencies:

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

yaml

Code:

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/models/song.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(),
  // ),
  // );
  runApp(
    MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (context) => LikedSongsModel()),
        // Add other providers if needed
      ],
      child: MyApp(),
    ),
  );
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return ChangeNotifierProvider(
      create: (context) => LikedSongsModel(),
      child: MaterialApp(
        title: 'Spotify',
        home: WelcomePage(), // Use your main screen here
      ),
    );
  }
}
```

Firebase_options.dart:

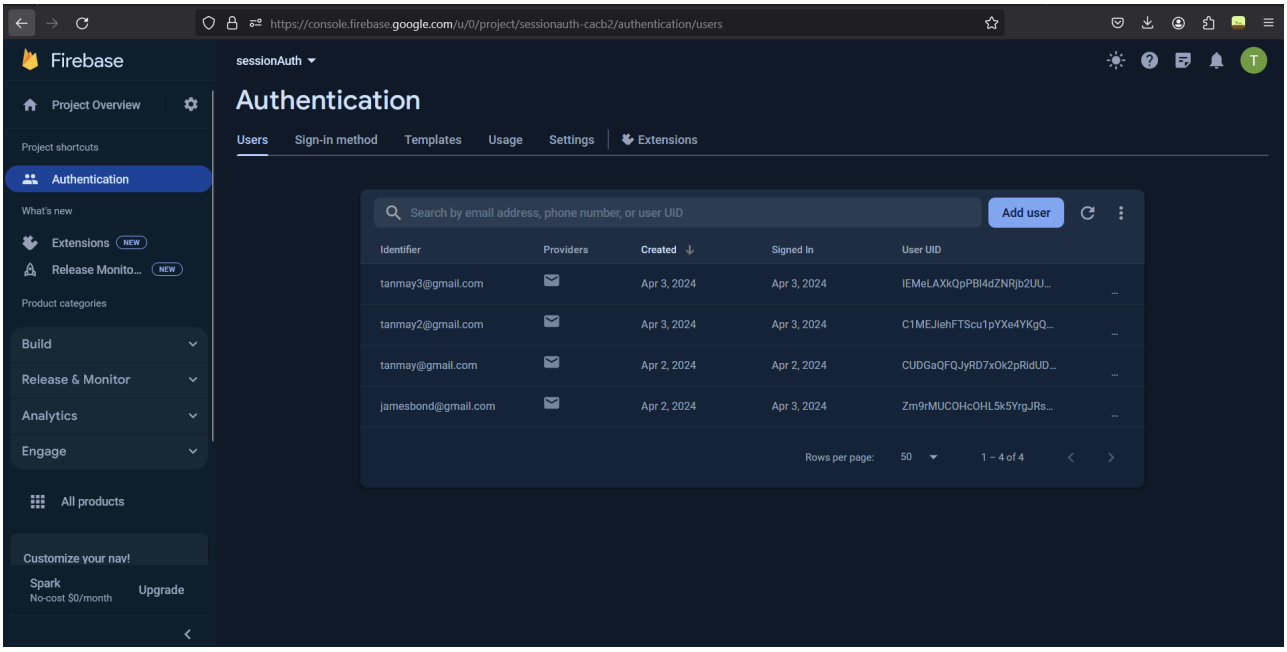
```
// File generated by FlutterFire CLI.
// ignore_for_file: lines_longer_than_80_chars,
avoid_classes_with_only_static_members
import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;
```

```dart
import 'package:flutter/foundation.dart'
    show defaultTargetPlatform, kIsWeb, TargetPlatform;

/// Default [FirebaseOptions] for use with your Firebase apps.
///
/// Example:
/// ```dart
/// import 'firebase_options.dart';
/// // ...
/// await Firebase.initializeApp(
/// options: DefaultFirebaseOptions.currentPlatform,
/// );
/// ```
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for web - '
        'you can reconfigure this by running the FlutterFire CLI
        again.',
      );
    }
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        return android;
      case TargetPlatform.iOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for ios - '
          'you can reconfigure this by running the FlutterFire CLI
          again.',
        );
      case TargetPlatform.macOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for macos -
```

```
  ' 'you can reconfigure this by running the FlutterFire CLI
  again.',
);
```

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

  static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'AIzaSyBCubxVCQHG3ieLoA11-3EzmohuAsMET8c',
    appId: '1:867583680837:android:81bfa85c82da3424599e2c',
    messagingSenderId: '867583680837',
    projectId: 'phone-authentication-ddb5c',
    storageBucket: 'phone-authentication-ddb5c.appspot.com',
  );
}
```

Firebase console:



Conclusion:

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

MAD and PWA Lab

Name: Tanmay Bhosale **Class: D15A** **Roll no:08**

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">
```

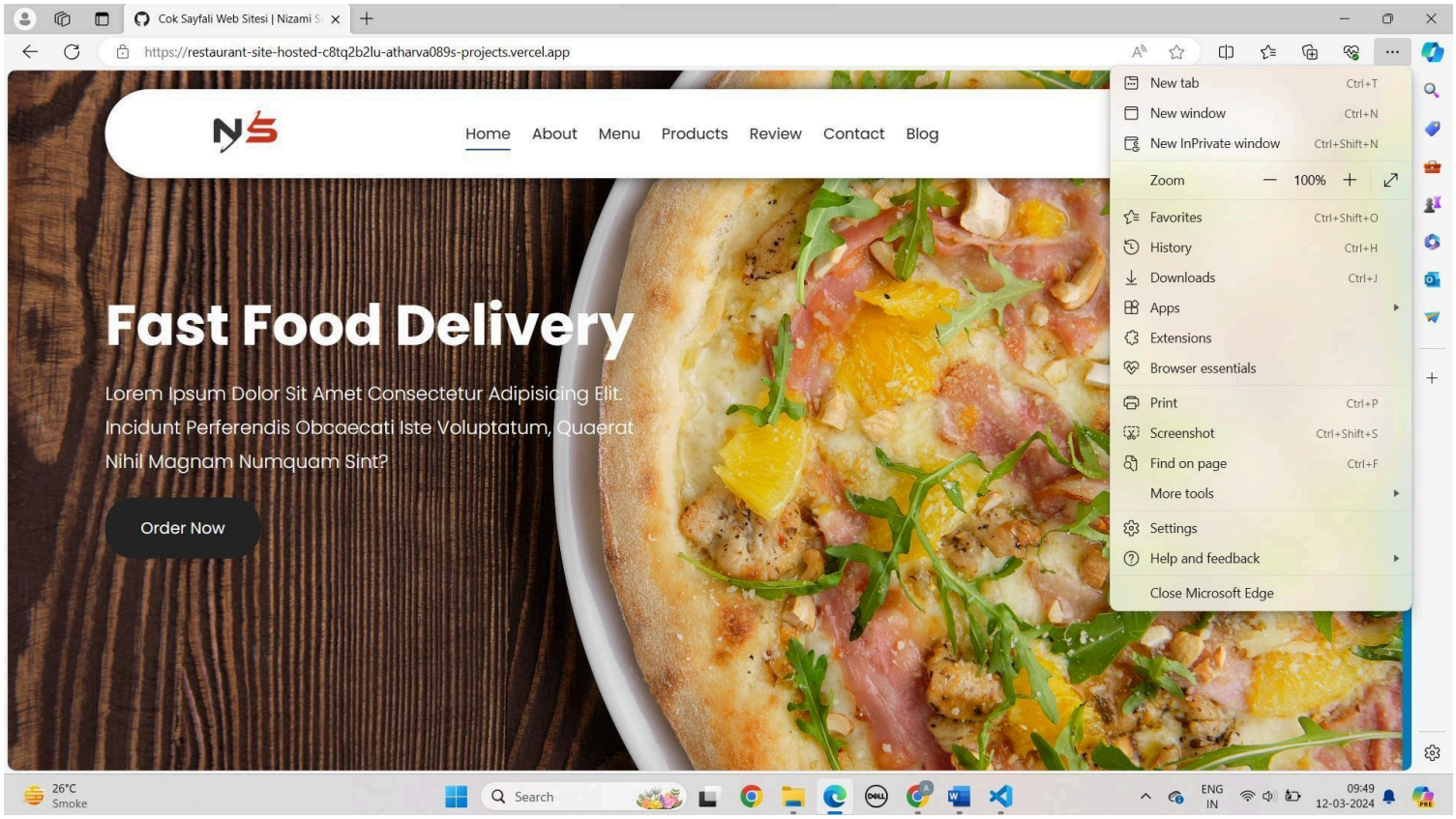
```
                <i class="fas fa-envelope"></i>
                <input type="email" placeholder="email">
            </div>
            <div class="inputBox">
                <i class="fas fa-phone"></i>
                <input type="number" placeholder="number">
            </div>
            <input 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">
```

how to make burgers
by admin / 10st may, 2020
Lorem ipsum dolor sit amet, consectetur adipisicing elit. Minus eos esse nesciunt cupiditate expedita.

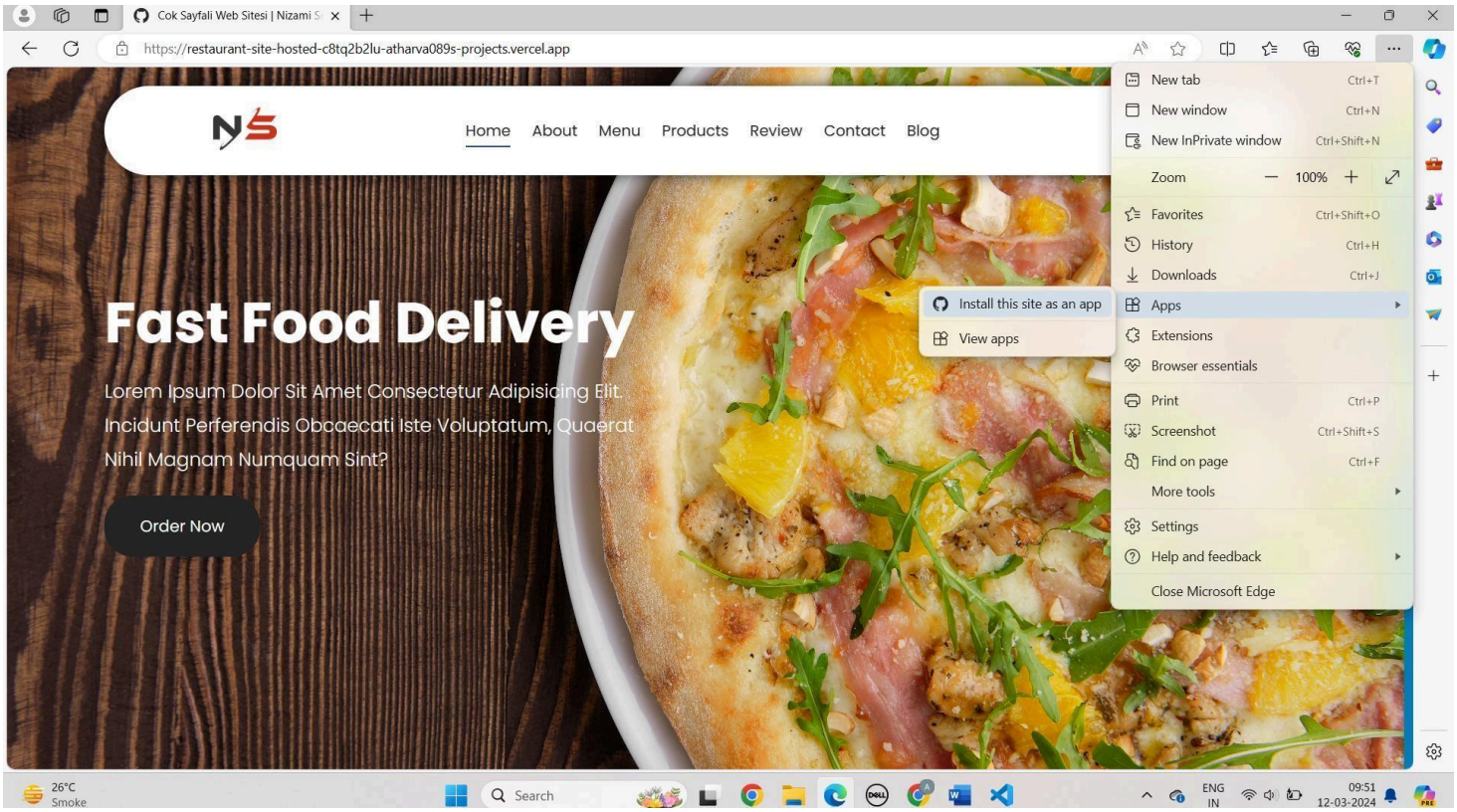
```
                <a href="#" class="btn">read more</a>
              </div>
            </div>
          </div>
        </section>
        <!_____________________________BLOG SECTION_>

        <!_____________________________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

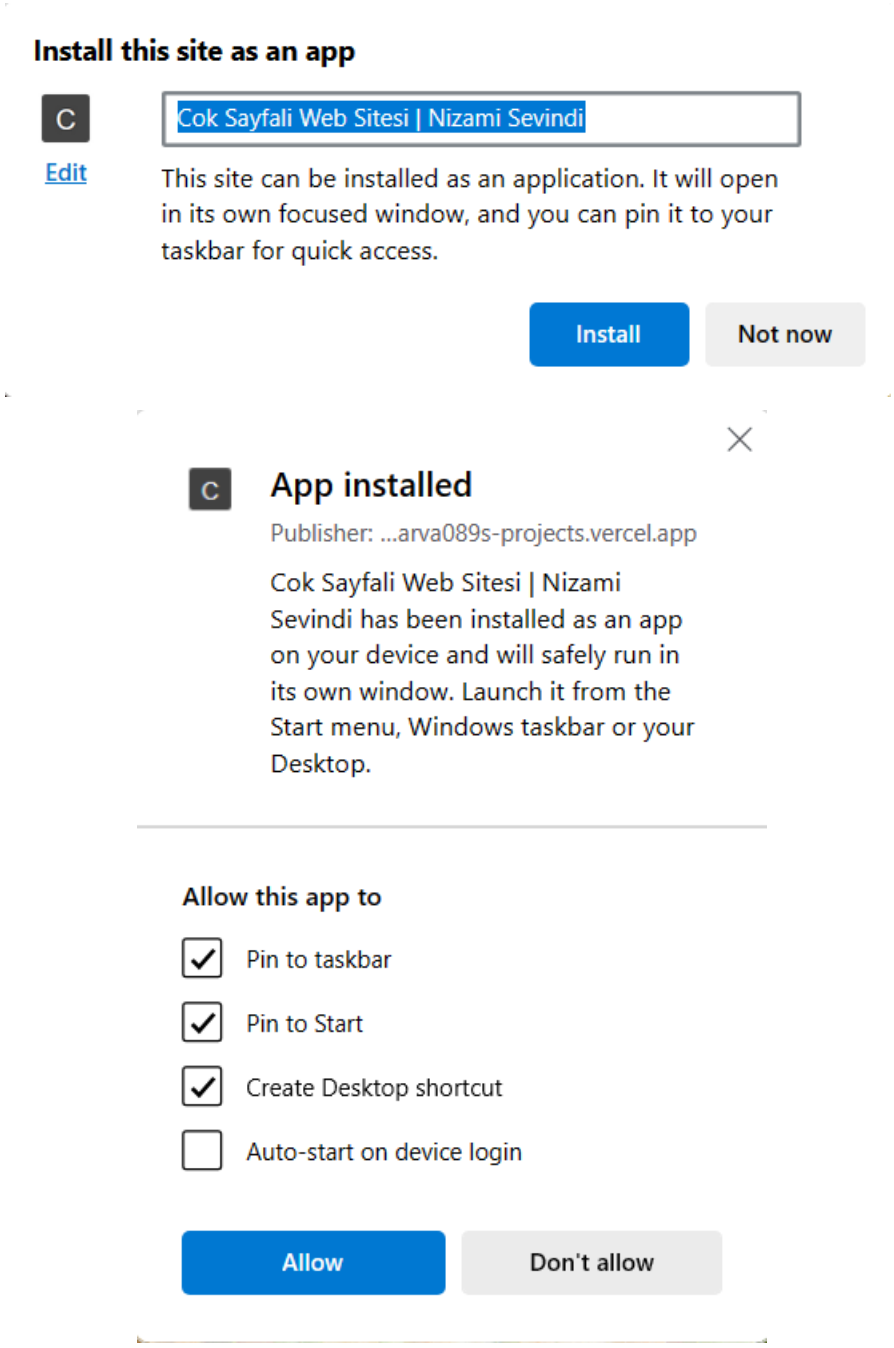
Open your hosted site on Microsoft Edge



Click on 3
dots click on
Apps
select install app option

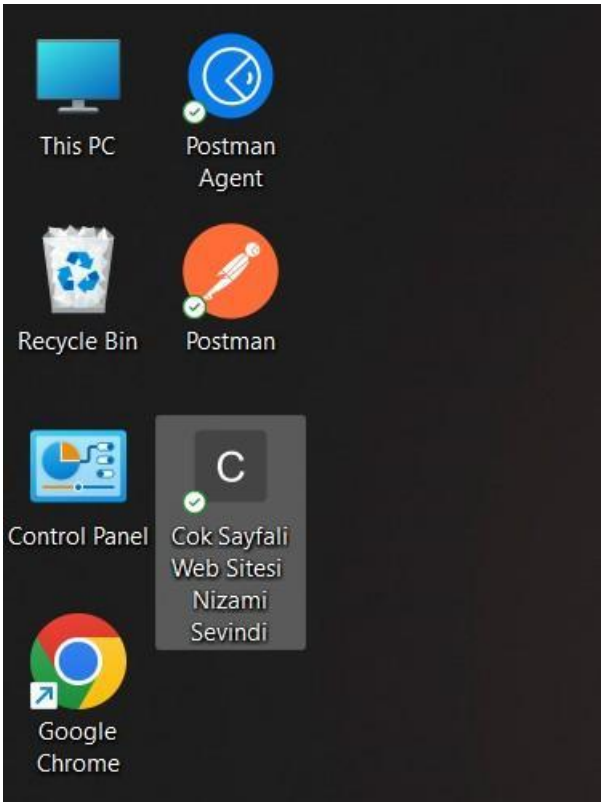


Click on Install



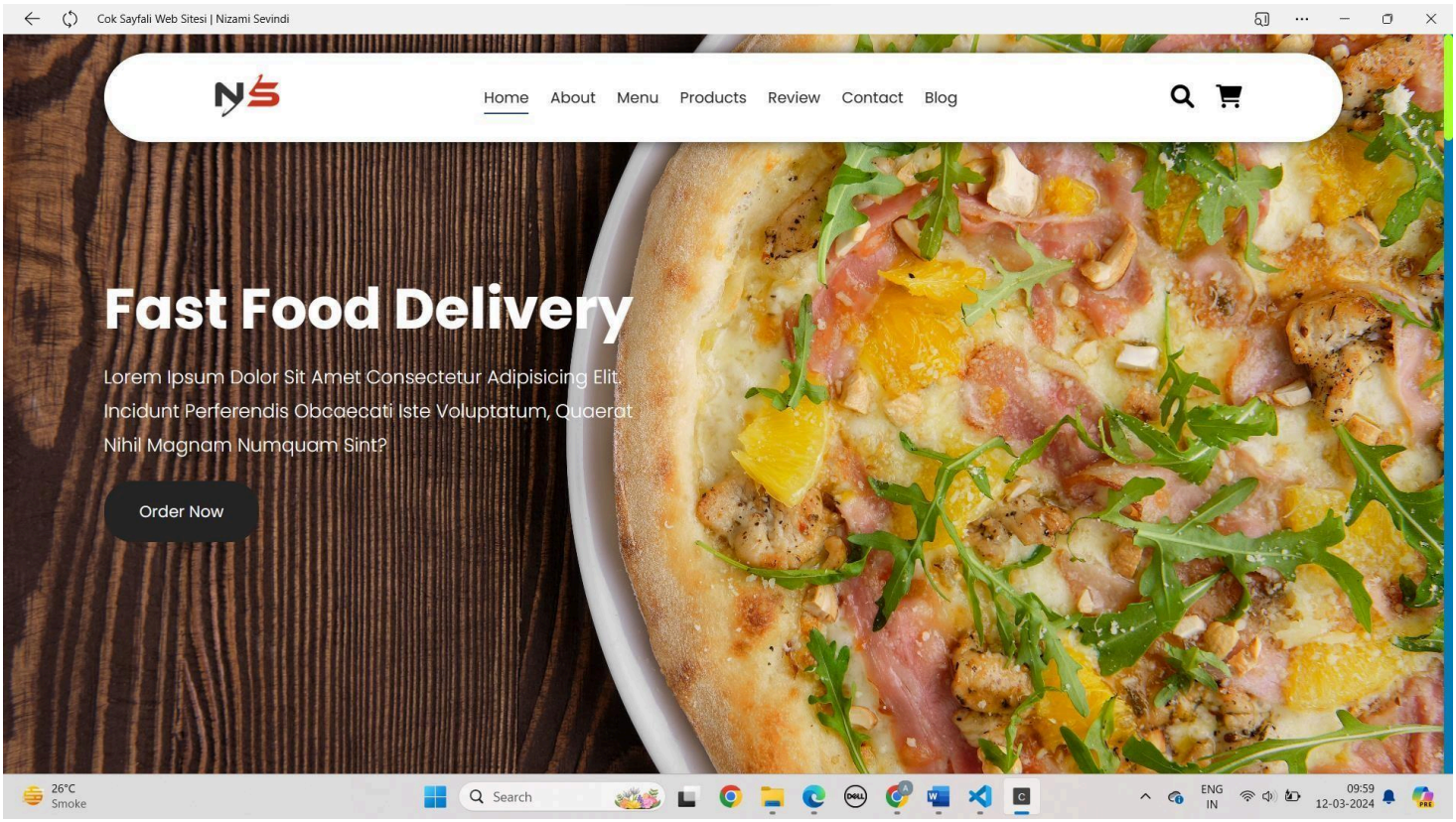
App icon will appear at the bottom

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 and PWA Lab

Name: Tanmay Bhosale Class: D15A Roll no:08

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-
KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQjoXtEKNosOWa
ZqXgel0g==" 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;
}
```

```
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:

```
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:

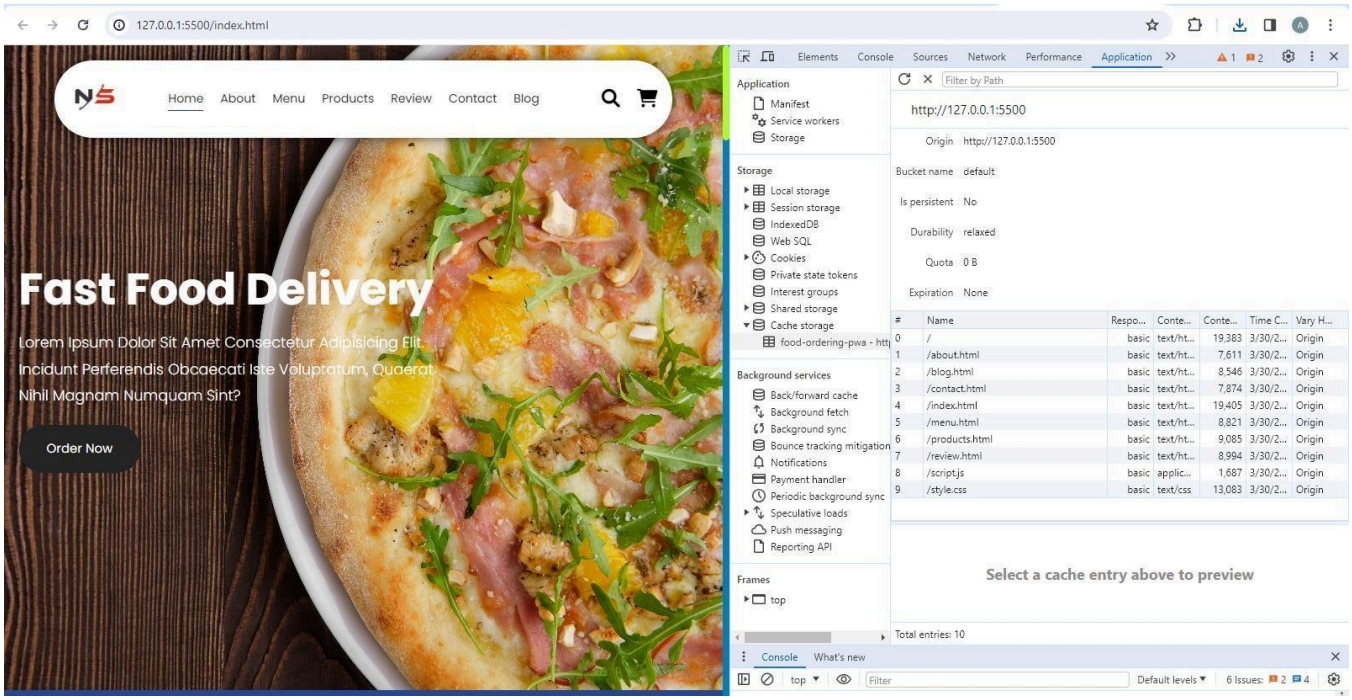
```
// service-worker.js


const cacheName = 'gfg-pwa-v1';
const assetsToCache = [
    '/',
    '/index.htm
    l',
    '/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);
                })
            );
        })
    );
});
```

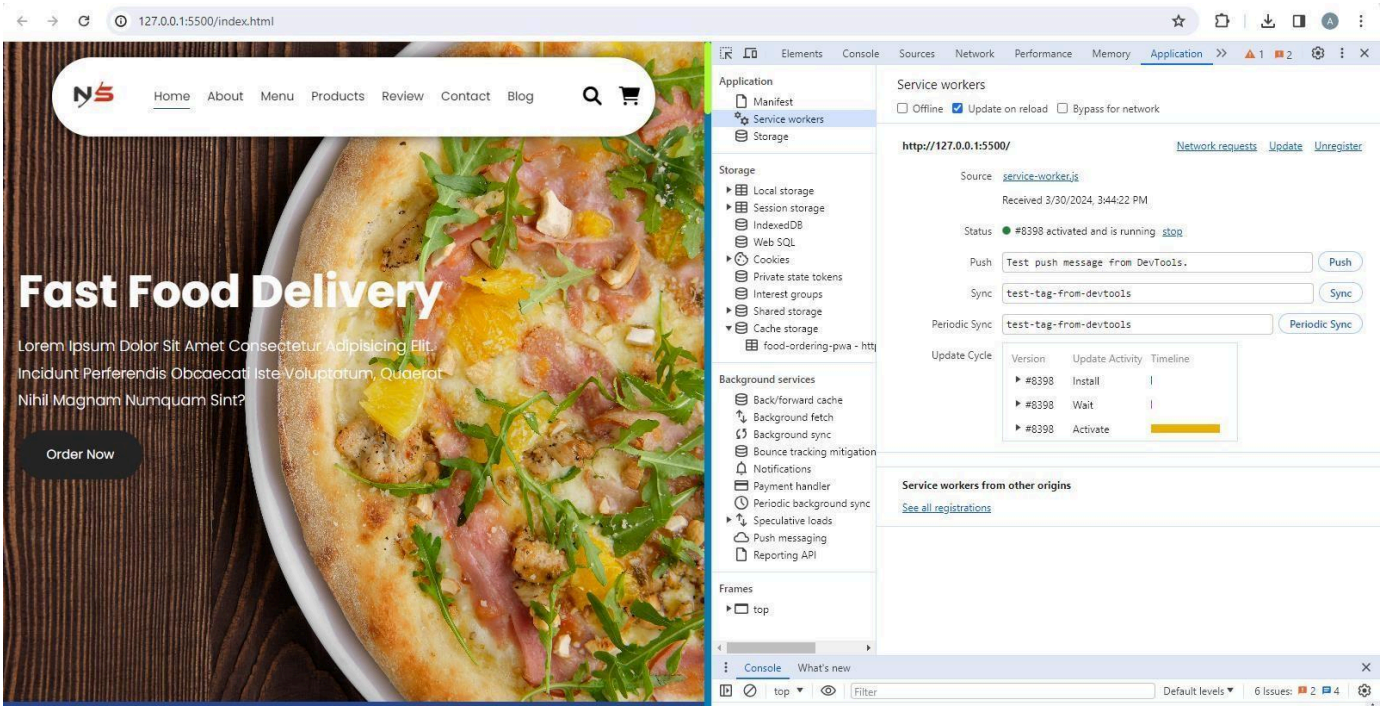
Steps for Execution:-

- 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 AND PWA LAB

Name: Tanmay Bhosale Class: D15A Roll no: 08

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.

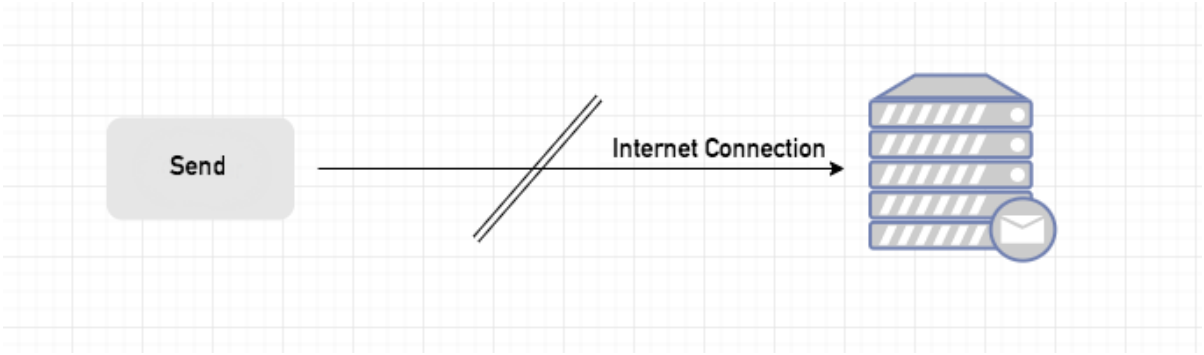
NetworkFirst - In this function, firstly we can try getting an updated response from the network, if this process completed successfully, the new response will be cached and returned. But if this process fails, we check whether the request has been cached before or not. If a cache exists, it is returned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

Sync Event

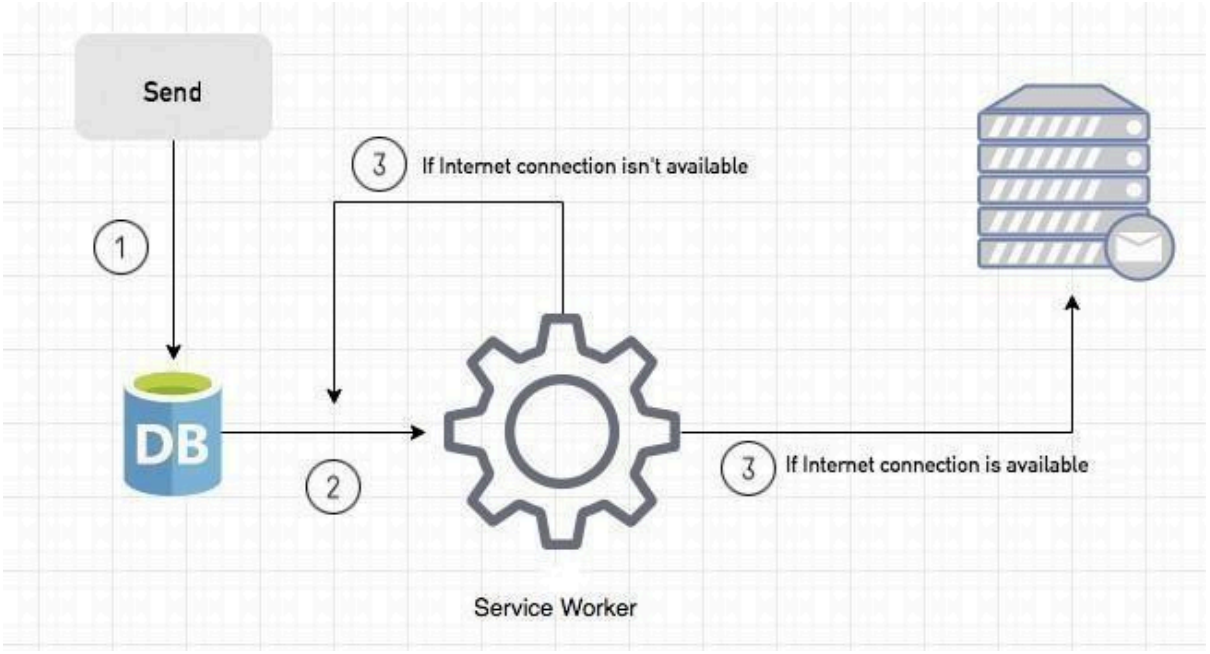
Background Sync is a Web API that is used to delay a process until the Internet connection is stable. We can adapt this definition to the real world; there is an e-mail client application that works on the browser and we want to send an email with this tool. Internet connection is broken while we are 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();
```

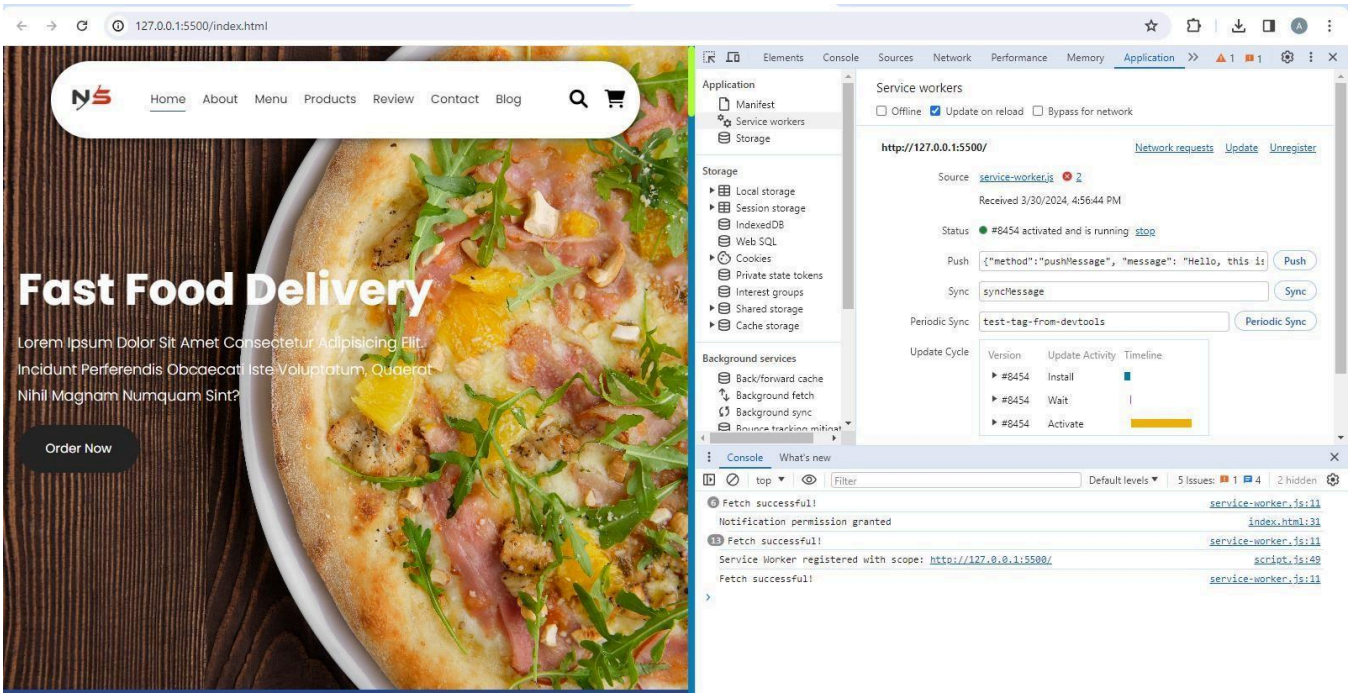
```
      if (data && data.method === "pushMessage") {
        console.log("Push notification sent");
        self.registration.showNotification("Twiggy", {
          body: data.message,
        });
      }
    } catch (error) {
      console.error("Error parsing push data:", error);
    }
  }
});
var 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"));
                                                  }
                                                })
                                                .
                                                  c
                                                  a
                                                  t
                                                  c
                                                  h
                                                  (
                                                  f
                                                  u
                                                  n
                                                  c
                                                  t
                                                  i
                                                  o
  });
};
```

```
n                                                e
(                                                c
e                                                t
r                                                (
r                                                e
o                                                r
r                                                r
)                                                o
{                                                r
r                                                )
e                                                ;
j                                              });
var returnFromCache = function (request) {
  return caches.open("offline").then(function (cache) {
    return cache.match(request).then(function (matching) {
      if (!matching || matching.status == 404) {
```

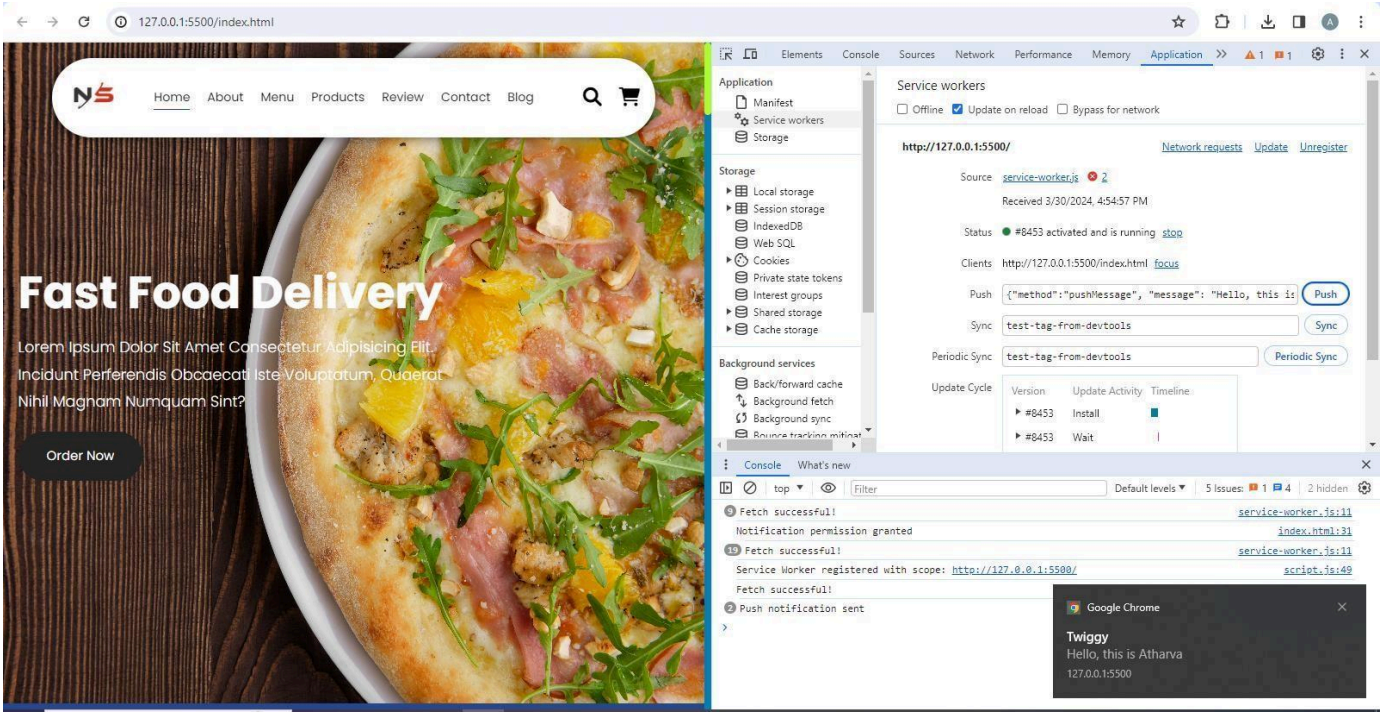
```
        return cache.match("offline.html");
      } else {
        return matching;
    }
  });
  });
};
var addToCache = function (request) {
  return caches.open("offline").then(function
    (cache) { return fetch(request).then(function
    (response) {
      return cache.put(request, response.clone()).then(function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

Output:

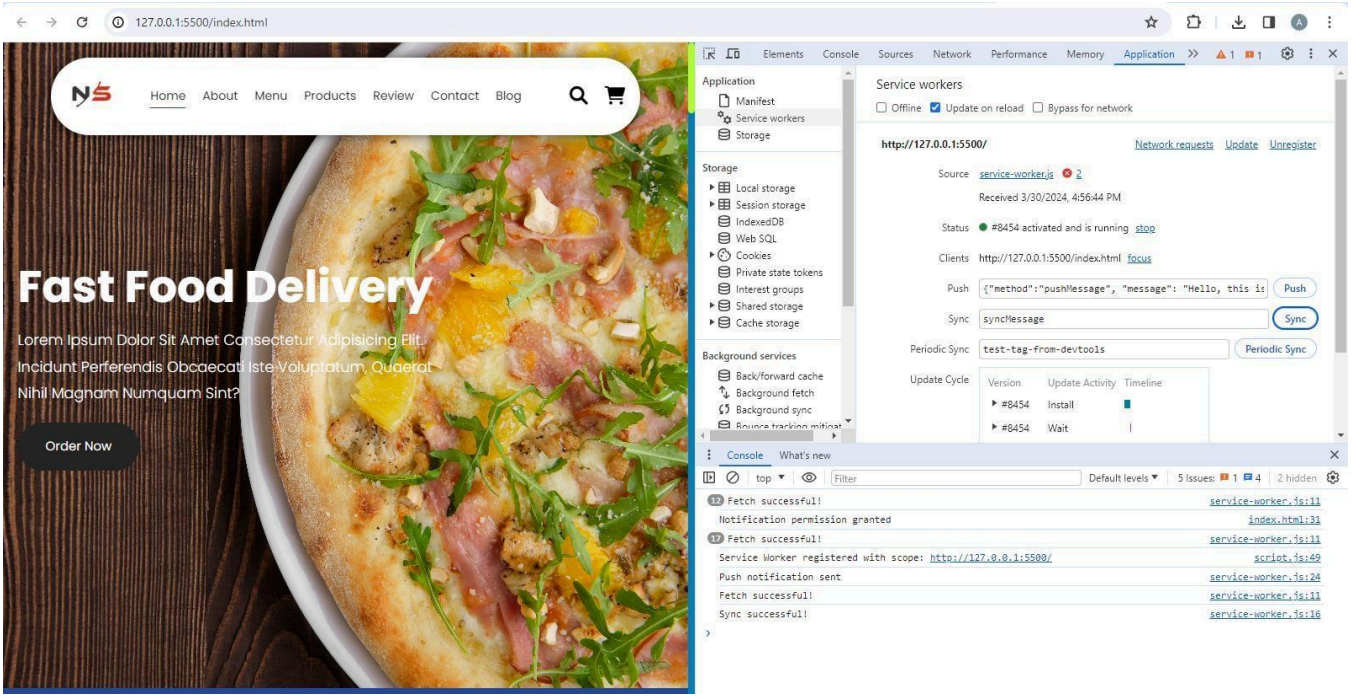
Fetch Event



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 and PWA Lab

Name: Tanmay Bhosale Class: D15A Roll no:08

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

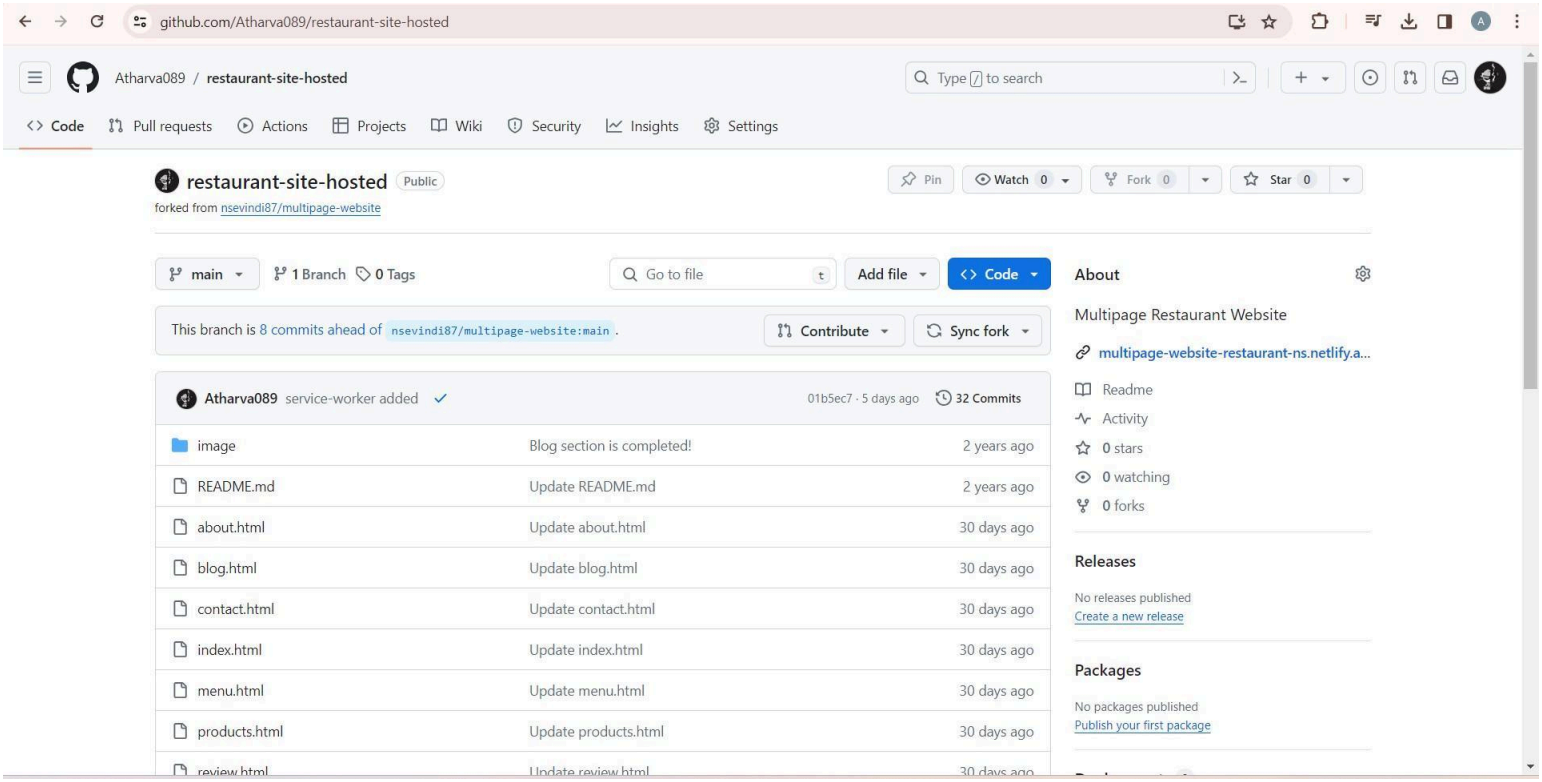
Pros

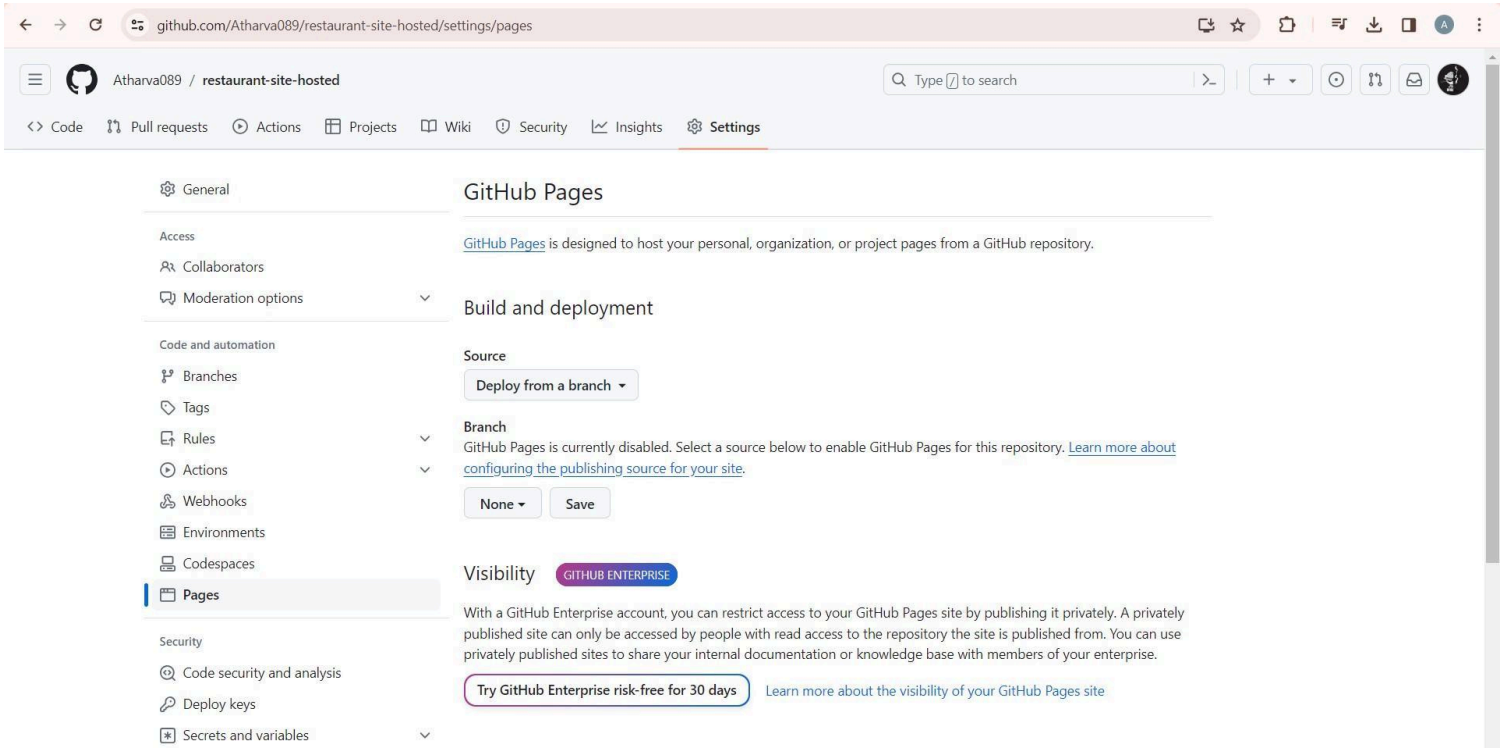
1. Hosted by Google. Enough said.
2. Authentication, Cloud Messaging, and a whole lot of other handy services will be available to you.
3. A real-time database will be available to you, which can store 1 GB of data.
4. You'll also have access to a blob store, which can store another 1 GB of data.
5.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

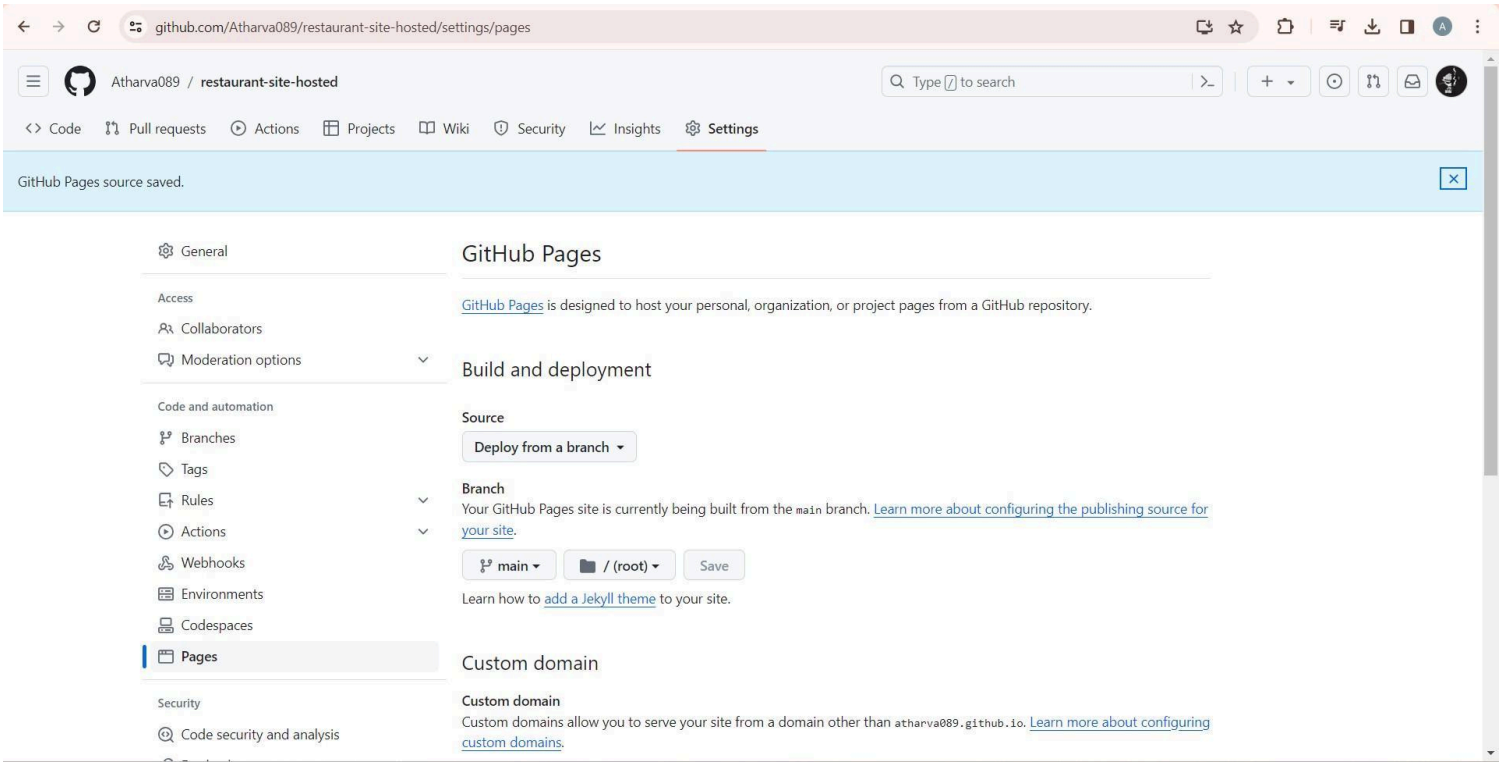
Cons

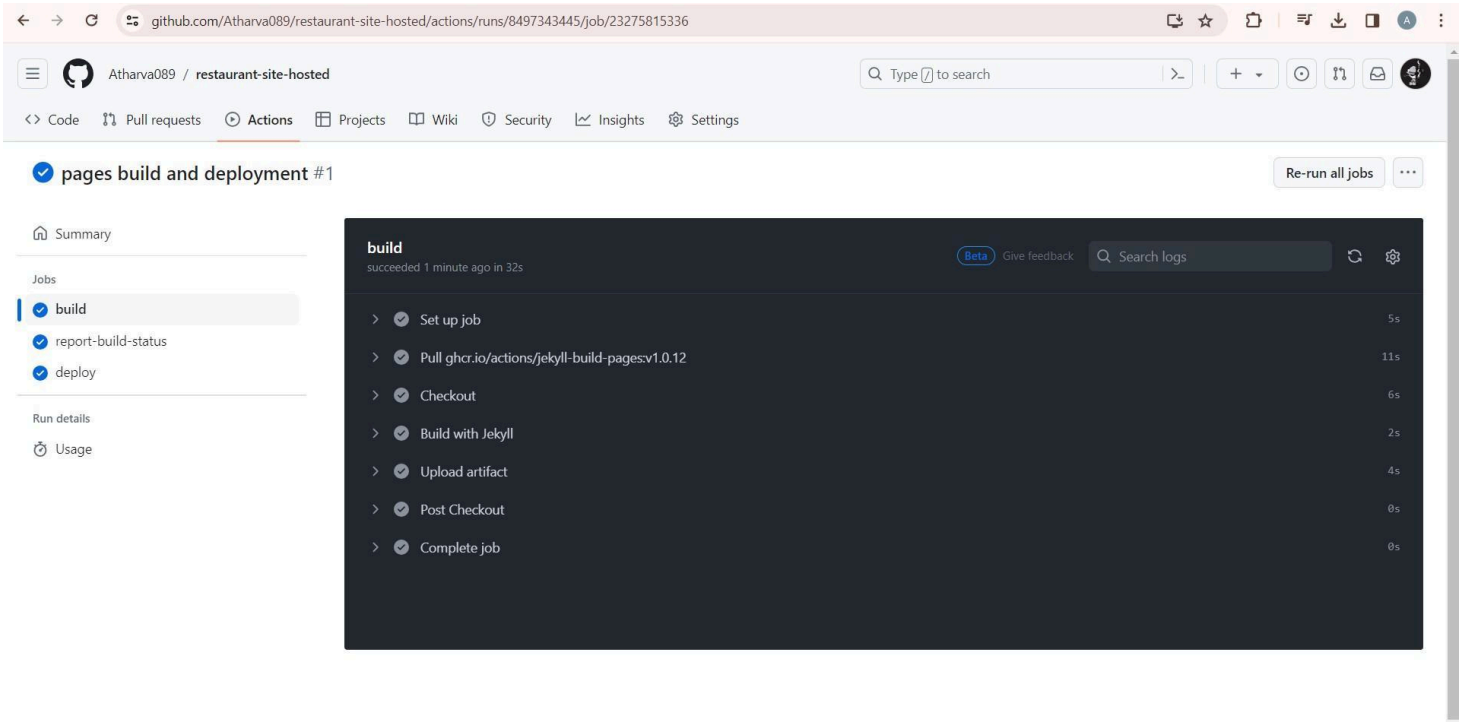
1. Only 10 GB of data transfer is allowed per month. But this is not really a big problem, if you use a CDN or AMP.
2. Command-line interface only.

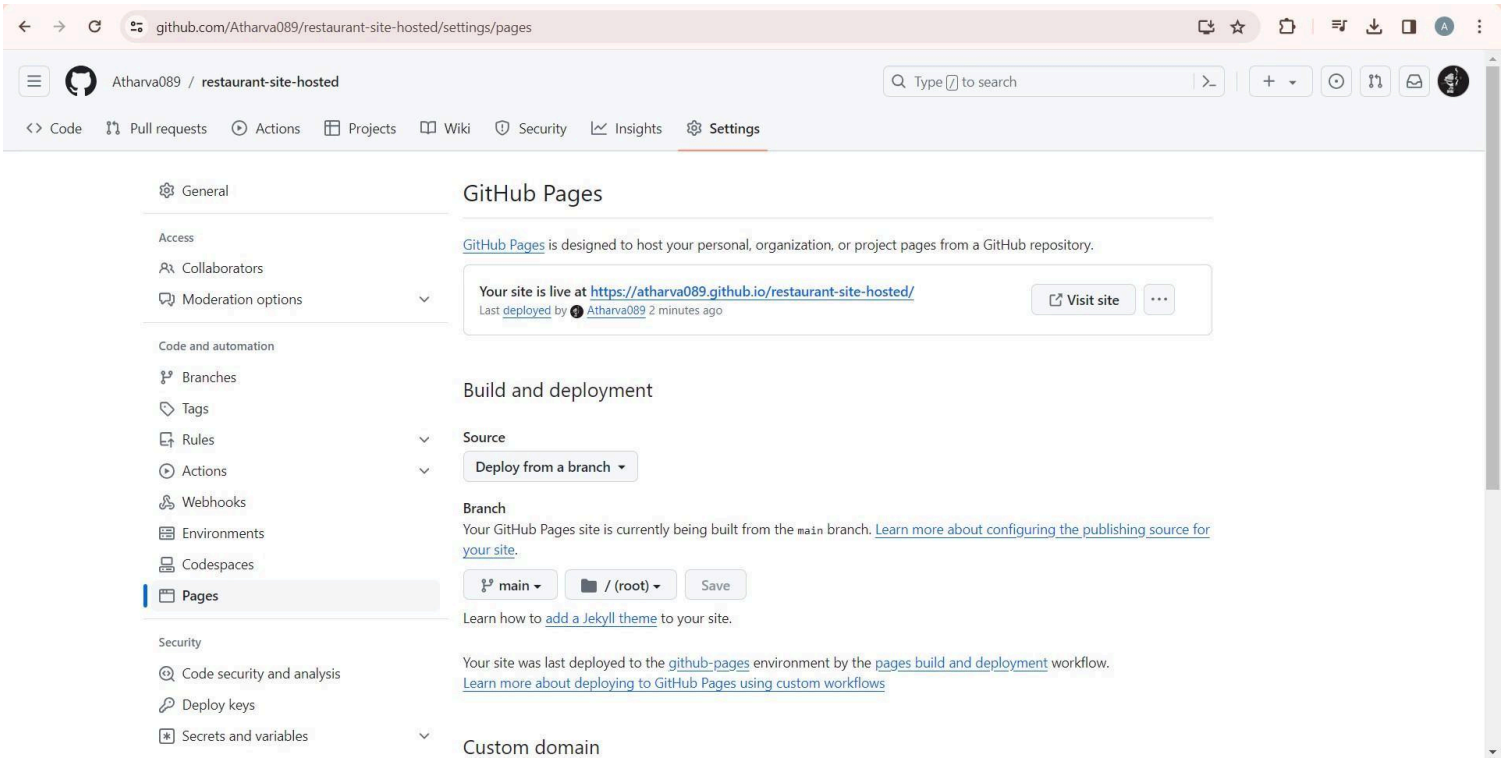
Implementation:

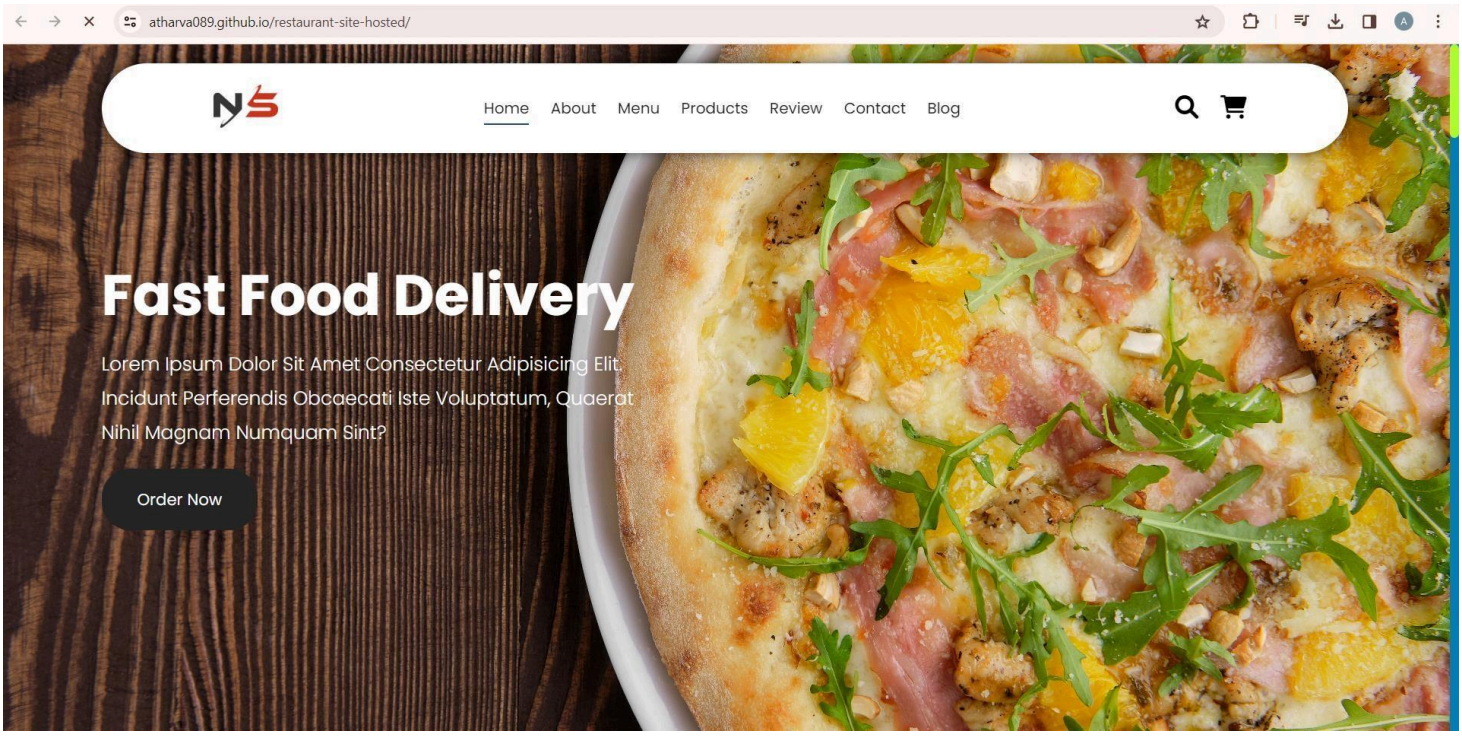





github.com/Atharva089/restaurant-site-hosted/settings/pages
Atharva089 / restaurant-site-hosted
Type / to search
Code
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
GitHub Pages source saved.
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
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 and PWA Lab

Name: Tanmay Bhosale **Class: D15A** **Roll no:08**

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

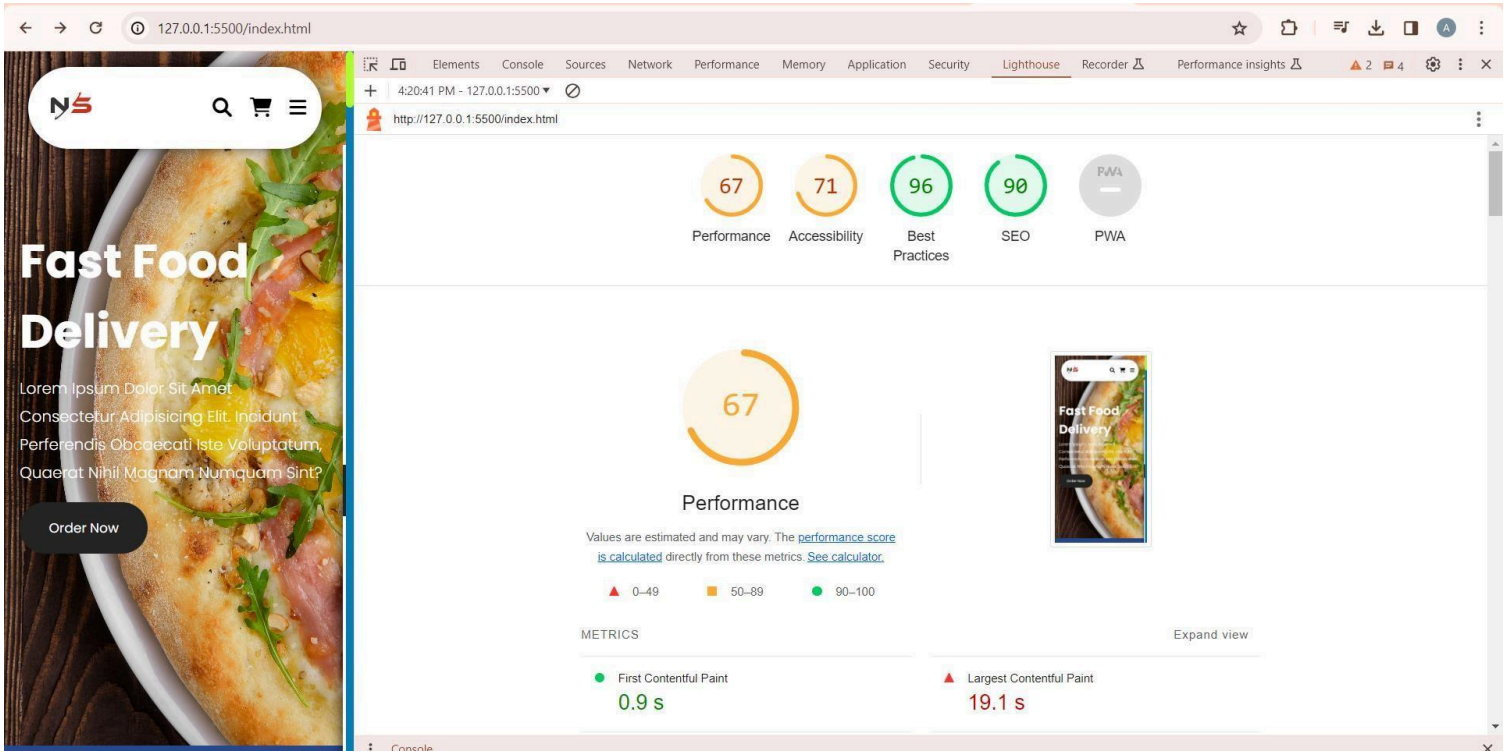
   Avoiding the use of deprecated code elements like tags, directives, libraries, etc. Password input with paste-into disabled

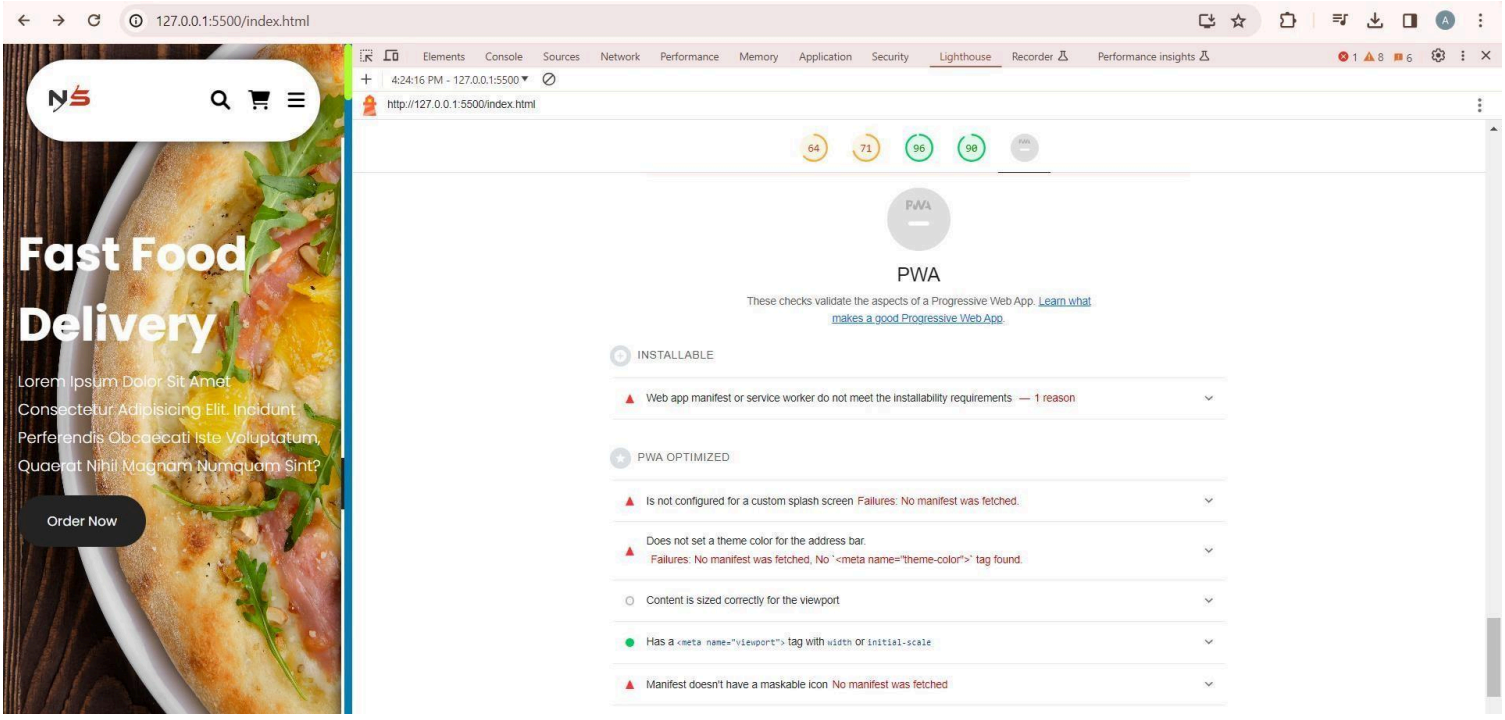
   Geo-Location and cookie usage alerts on load, etc.

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

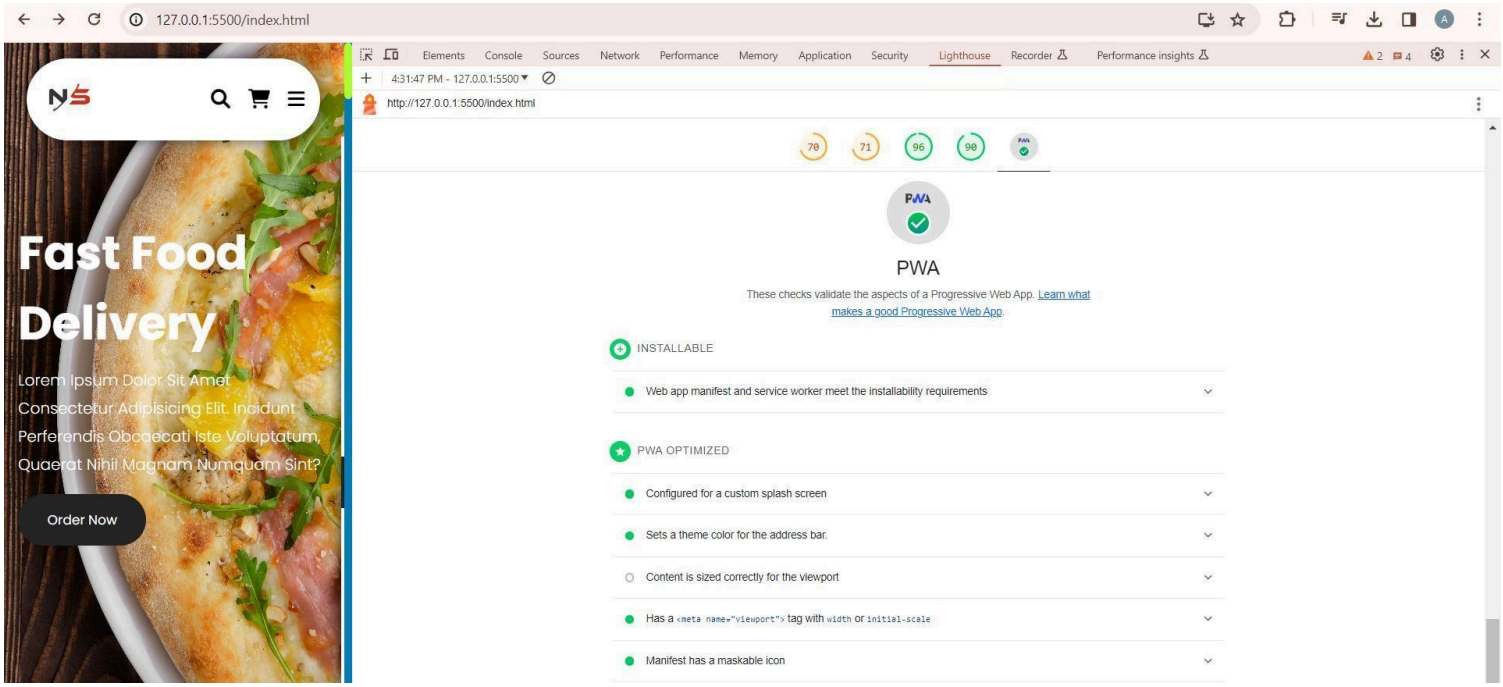
For theme color add a meta tag in index.html-

<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