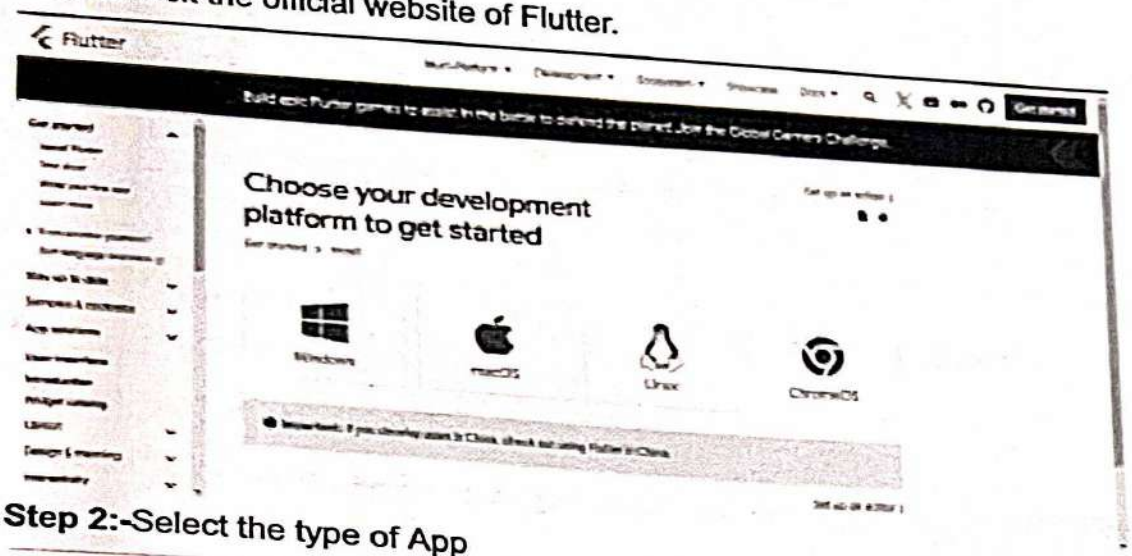
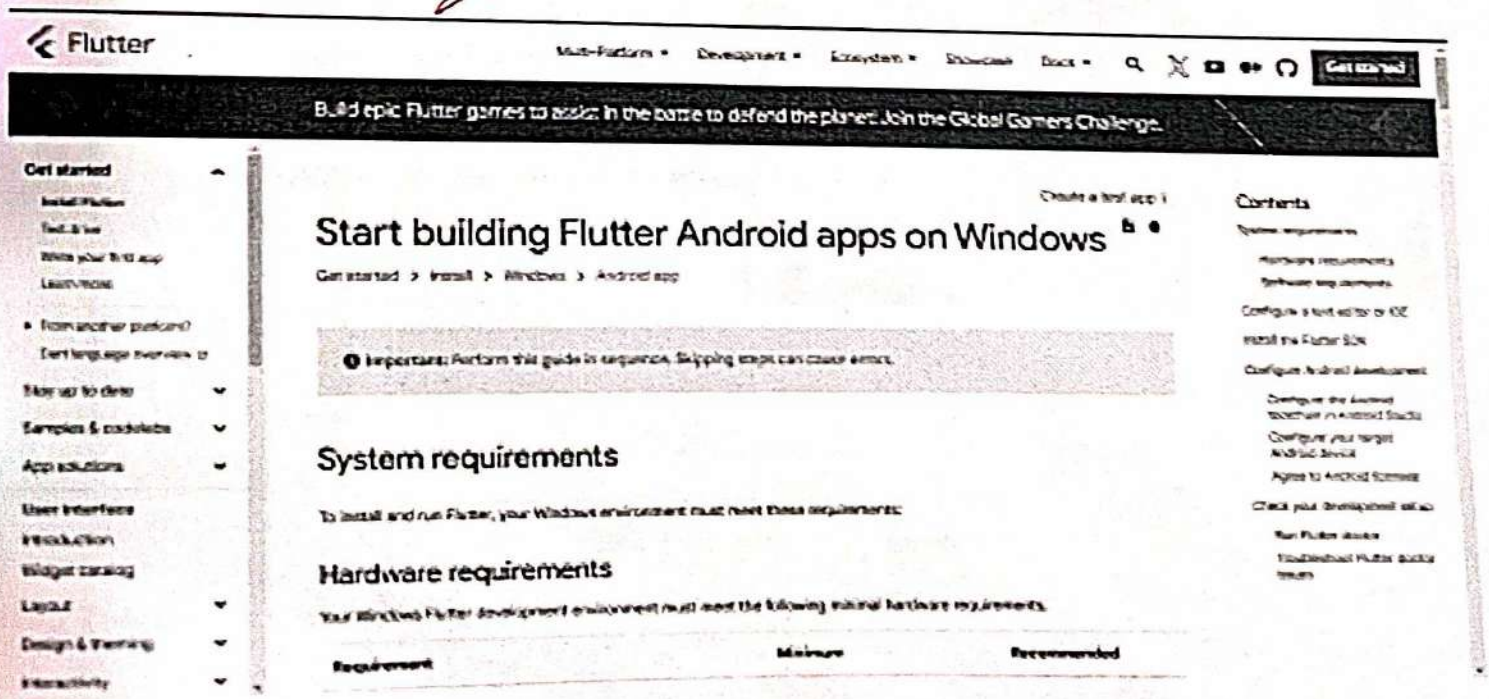
OUTPUT :-

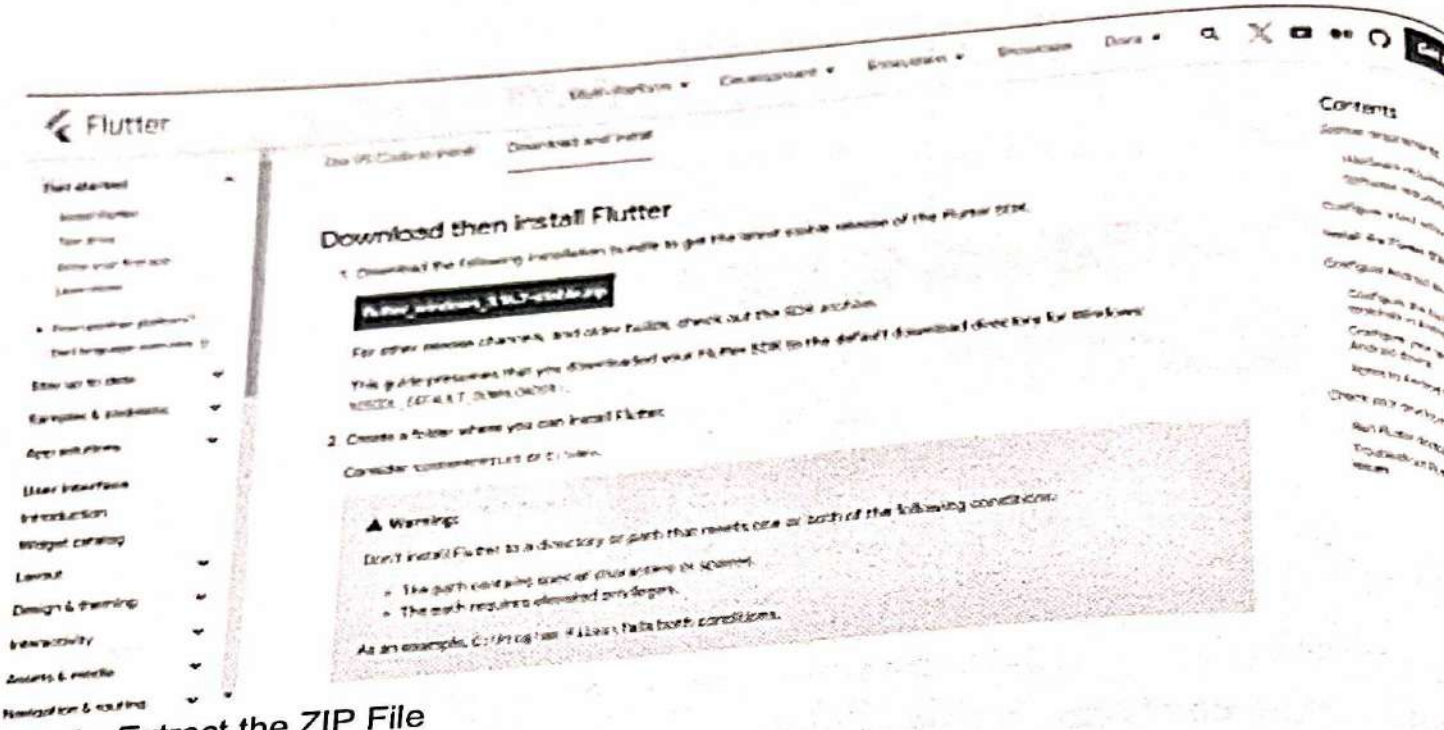
Step 1:-Visit the official website of Flutter.



Step 2:-Select the type of App

Step 3:- Download Flutter SDK

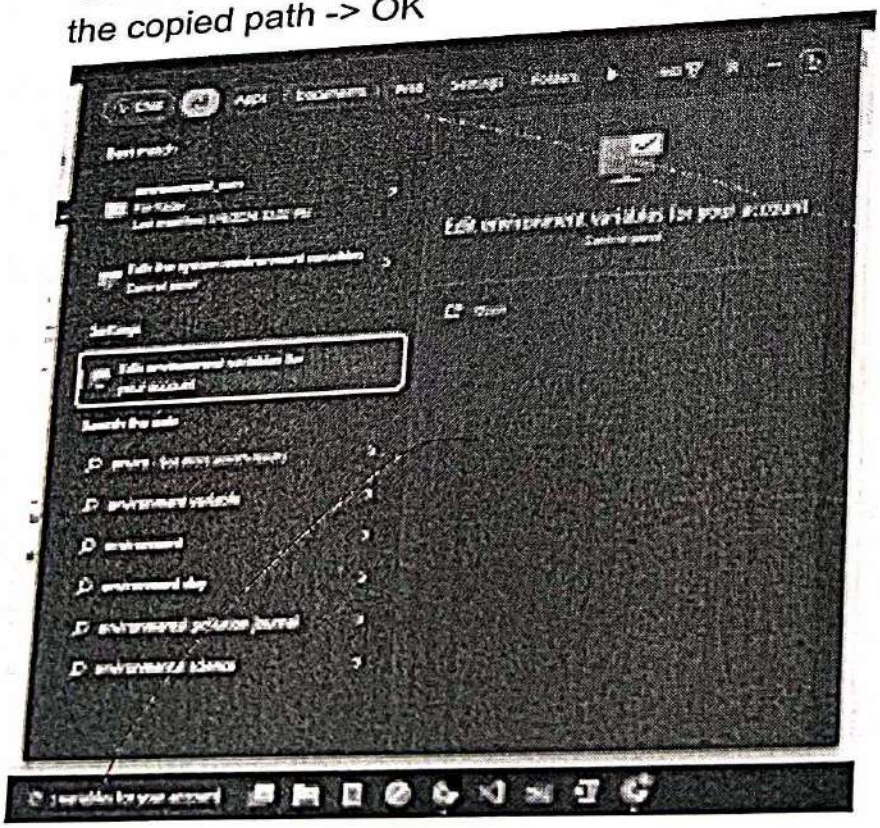


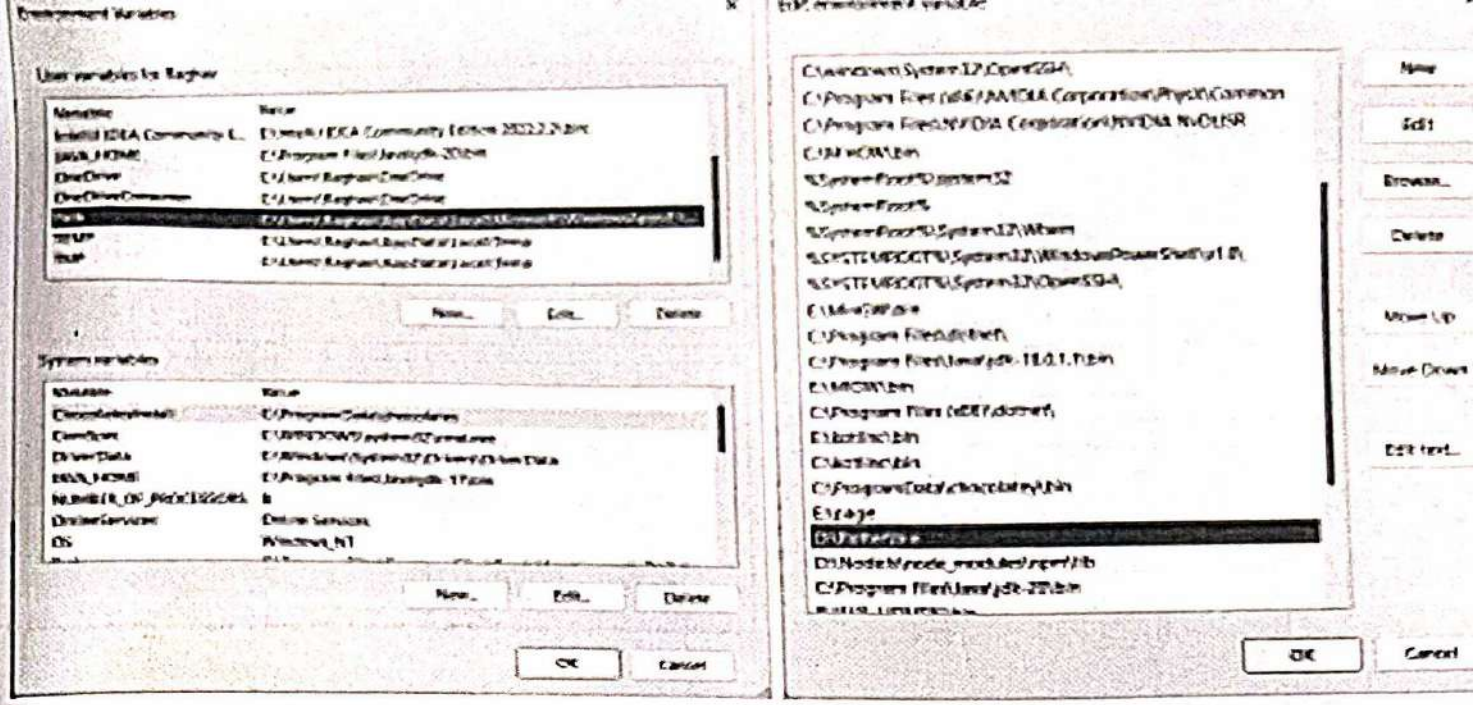


Step 4:- Extract the ZIP File

Step 5:-Add Flutter SDK path to environment variables

Go to extracted file -> select Bin Path -> search for environment variables -> Edit Path -> the copied path -> OK

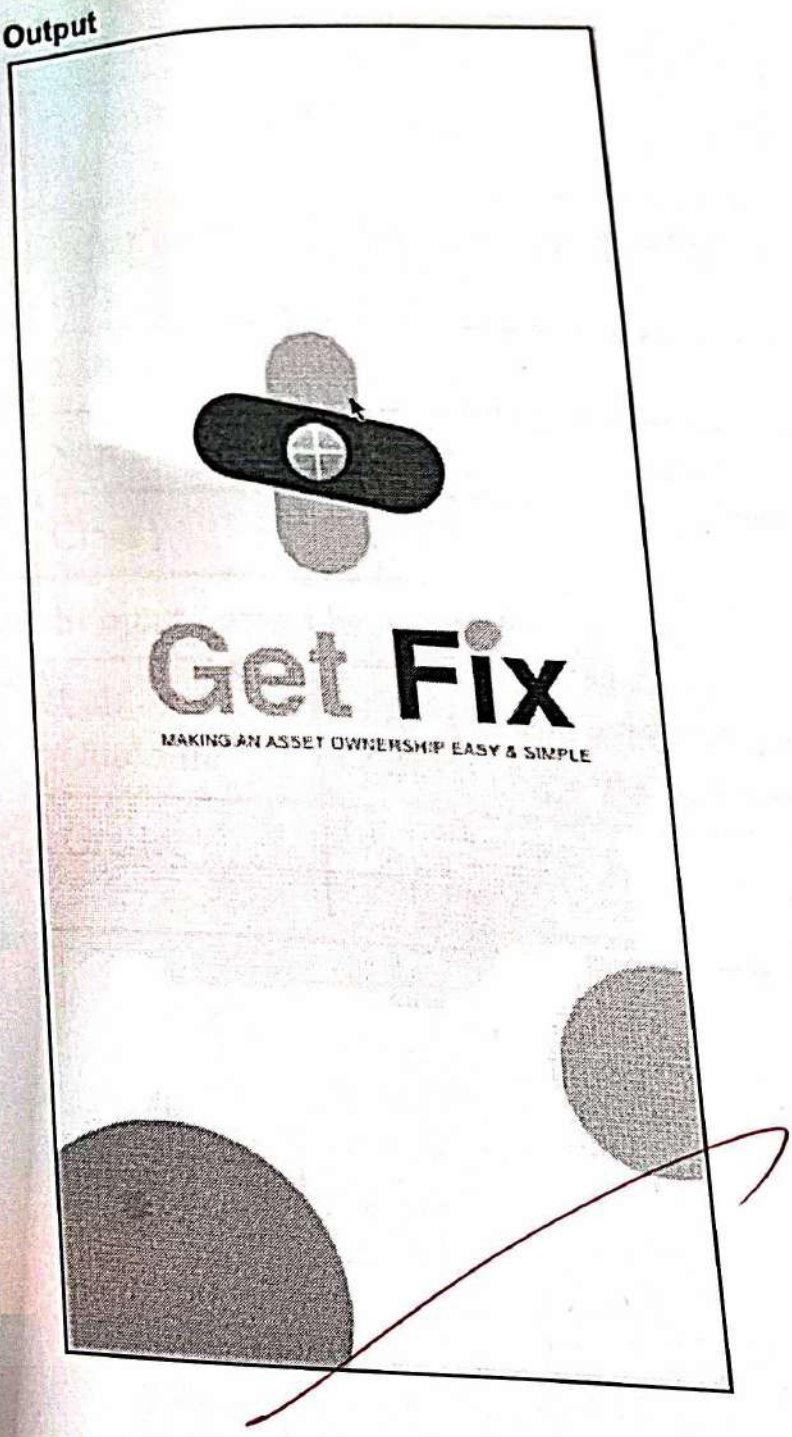


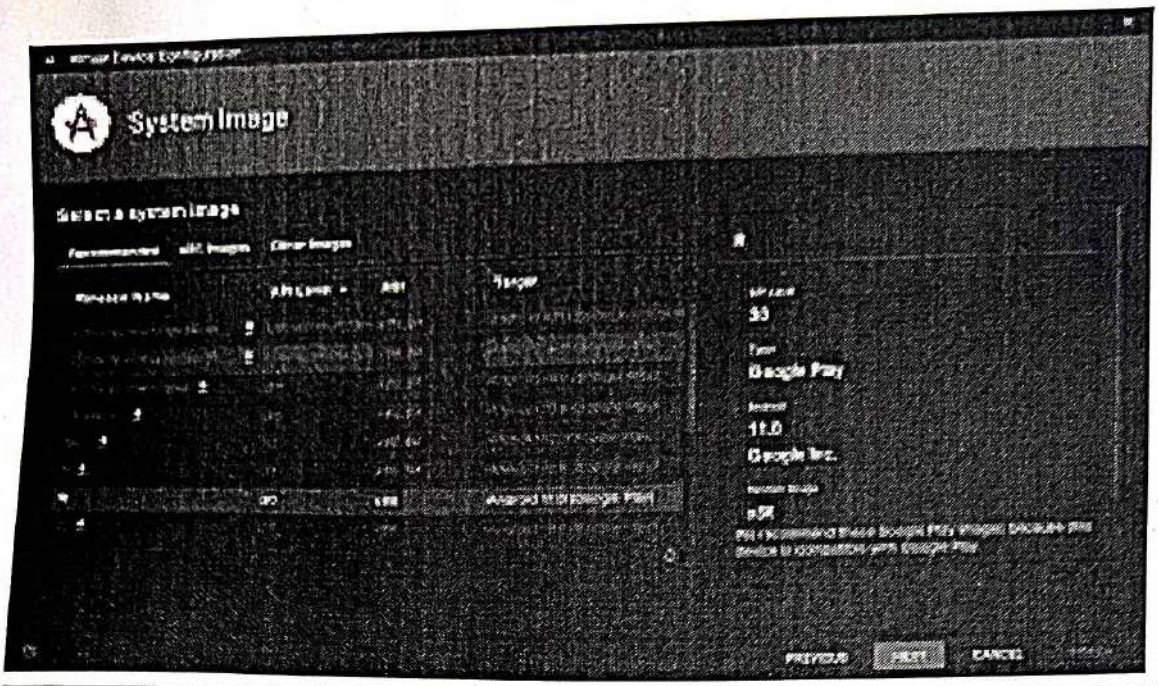


Step 6:- Visit Official Site for Android Studio

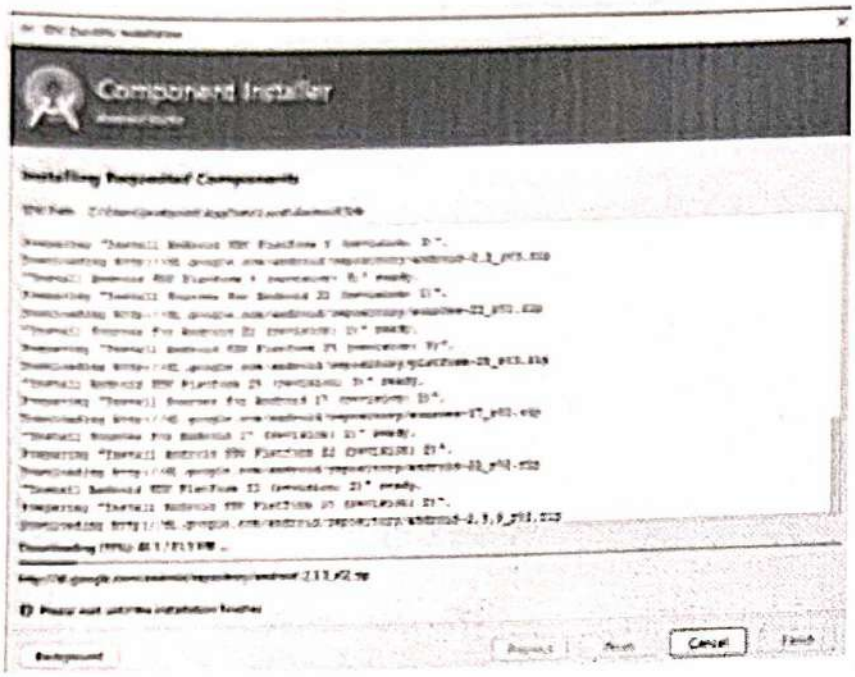
Step 7:-Install Android SDK

Output



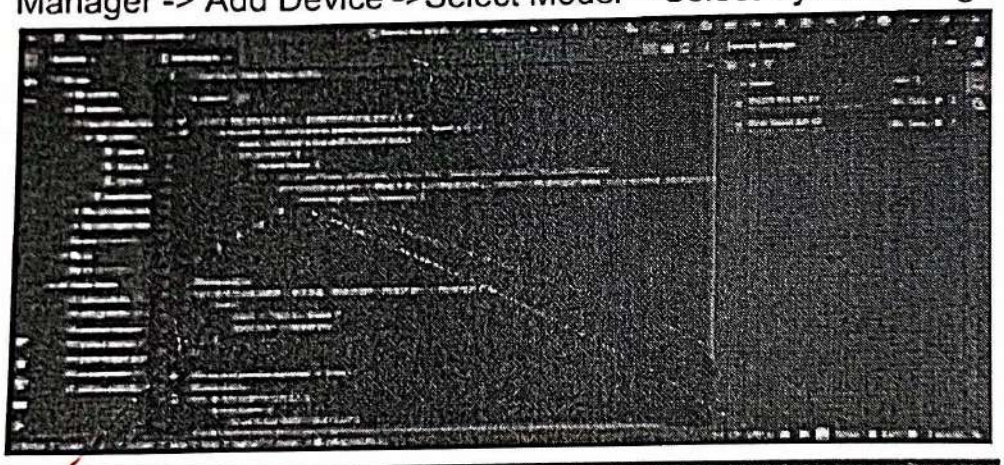

System Image
Google Play
11.0
PREVIOUS
NEXT
CANCEL

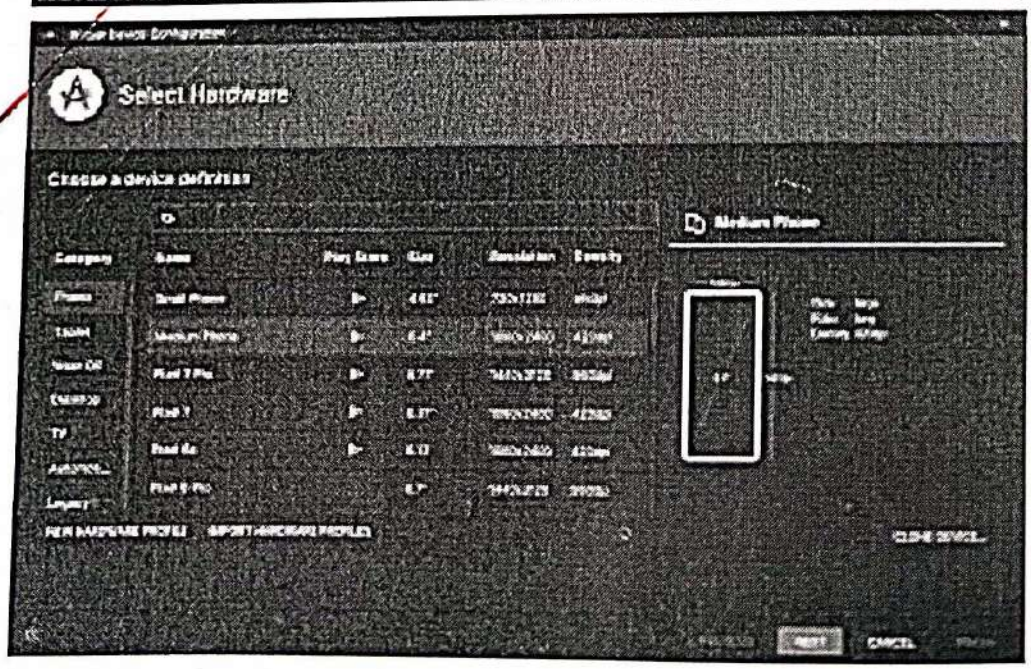
Android Virtual Device (AVD)
AVD Name
CANCEL
FINISH

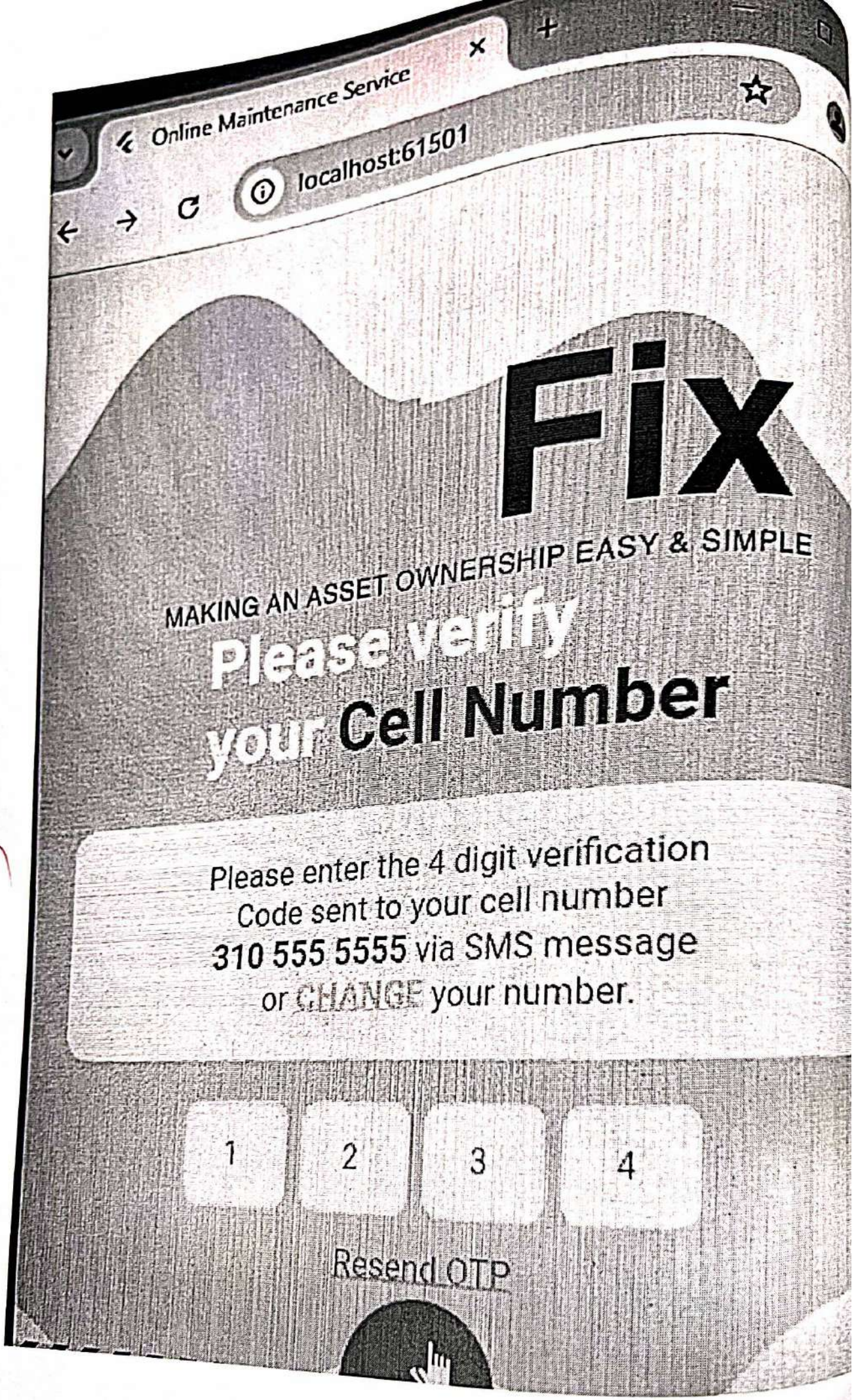


Step 8:-Accept Android License

Step 9:-Install AVD (Android Virtual Device) from Android Studio Device Manager -> Add Device ->Select Model ->Select System Image - > Finish






Online Maintenance Service
localhost:61501
Fix
MAKING AN ASSET OWNERSHIP EASY & SIMPLE
Please verify
your Cell Number
Please enter the 4 digit verification
Code sent to your cell number
310 555 5555 via SMS message
or CHANGE your number.
1
2
3
4
Resend OTP

Online Maintenance Service

localhost:61501

Get Fix

MAKING AN ASSET OWNERSHIP EASY & SIMPLE

SignUp

Sanika

Mehetre

abc@gmail.com

...

Type here to search

:48

0.00 KB/s 5G 99%

ome Services

elcome, abcd@gmail.com

Pest Control

Plumbing

Home Cooking

Online Maintenance Service

localhost:61501

MAKING AN ASSET OWNERSHIP EASY & SIMPLE

Sign In

Email

Password

SIGN IN

Forgot Password | OTP LOGIN

Or Sign In with

SIGNUP

Type here to search

Online Maintenance Service
localhost:61501
Get Fix
MAKING AN ASSET OWNERSHIP EASY & SIMPLE
Sign In
abc@gmail.com
SIGN IN
Forgot Password | OTP LOGIN
Or Sign In with
Type here to search

Online Maintenance Service

localhost:61501

Fix

MAKING AN ASSET OWNERSHIP EASY & SIMPLE

You Account Has Been Verified Successfully

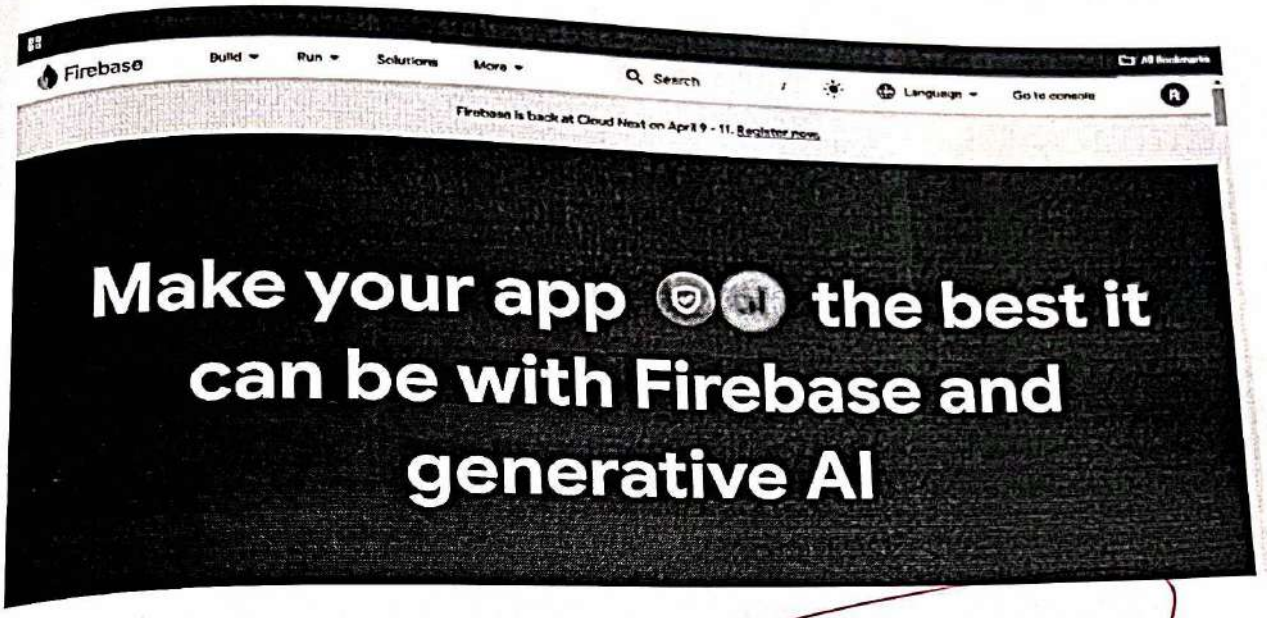
Type here to search

Name: Sanika Mehetre
Div: D15B / 34

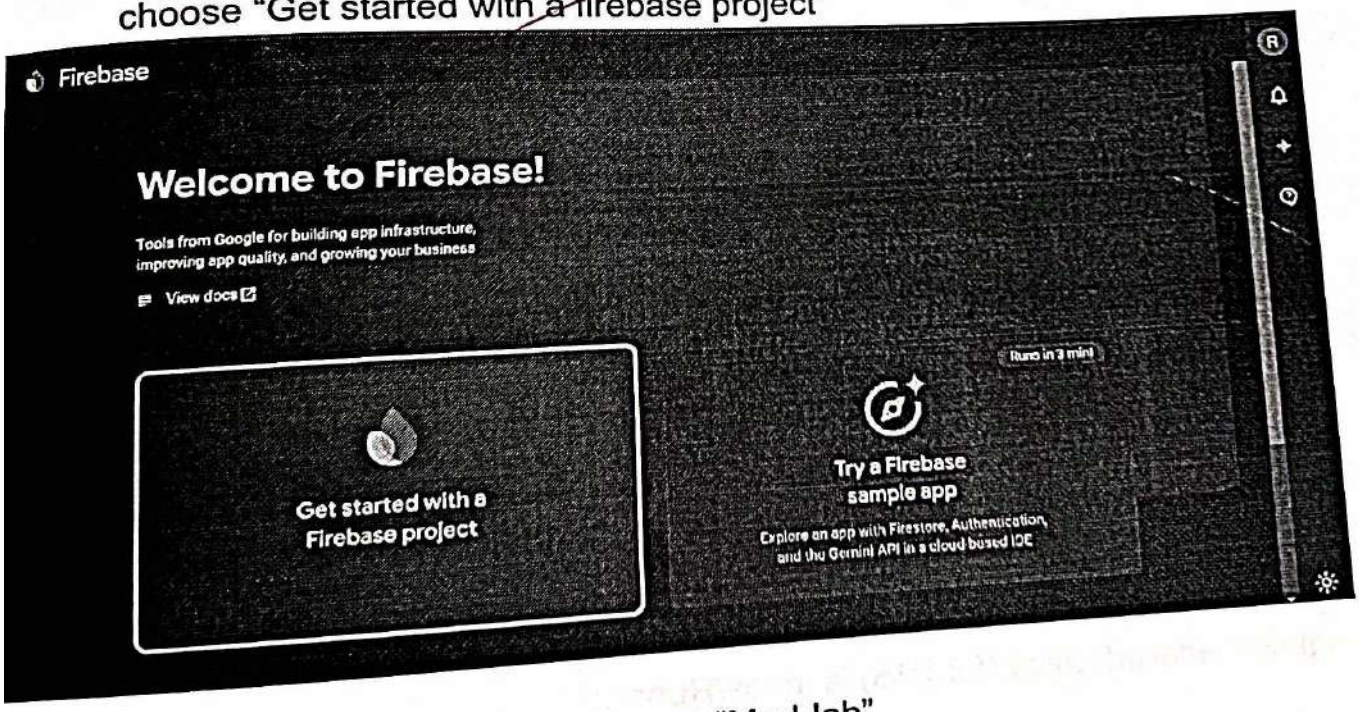
Aim: To setup firebase with flutter.

Steps:

1) Go to Firebase official website



2) Scroll down and we'll see a "Get started" button. Click on it. We can see the following, choose "Get started with a firebase project"



3) Name your project. I have named "Mad-lab"

Online Maintenance Service

localhost:61501

Book Your Service

Service at

Udhana

Udhna Taluka

Dindoli

Chikas Taluka

BOTTOM OVERFLOWED BY 10.0 PIXELS

Hygiene

Electrical

Appliances

Plumbing

AC Repair

Per Hour Charges:$20

Note: Part price is additional

Enter job description for service

Service Later

Book Now

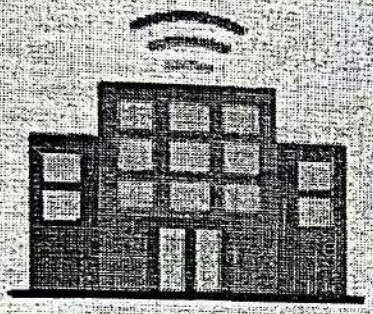
Type here to search

Online Maintenance Service

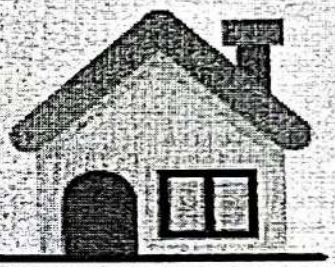
localhost:61501

Get Fix

Hi Choose Your Service Area



Business
Organisation



Home
Personal

Search for service ...

Current Location:
Service Location near - Jaipur 302019

Late

Search Now

> firebase_auth
> firebase_core
> firebase_storage
> native_assets
> path_provider_android

thriftstore_listings.dart U
firebase_options.dart U
main.dart M
> linux

analysis_options.yaml A
firebase.json U
mpl_lab.iml

- Ensure the firebase_core package is initialized correctly.

6) Initialize Firebase in main.dart:

```
lib > main.dart
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'pages/IntroPage.dart';

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

7) Run the following commands to add the packages
flutter pub add firebase_auth
flutter pub add cloud_firestore
flutter pub add firebase_storage
flutter pub add firebase_messaging

8) Android Configuration:

Ensure google-services.json is in android/app/.

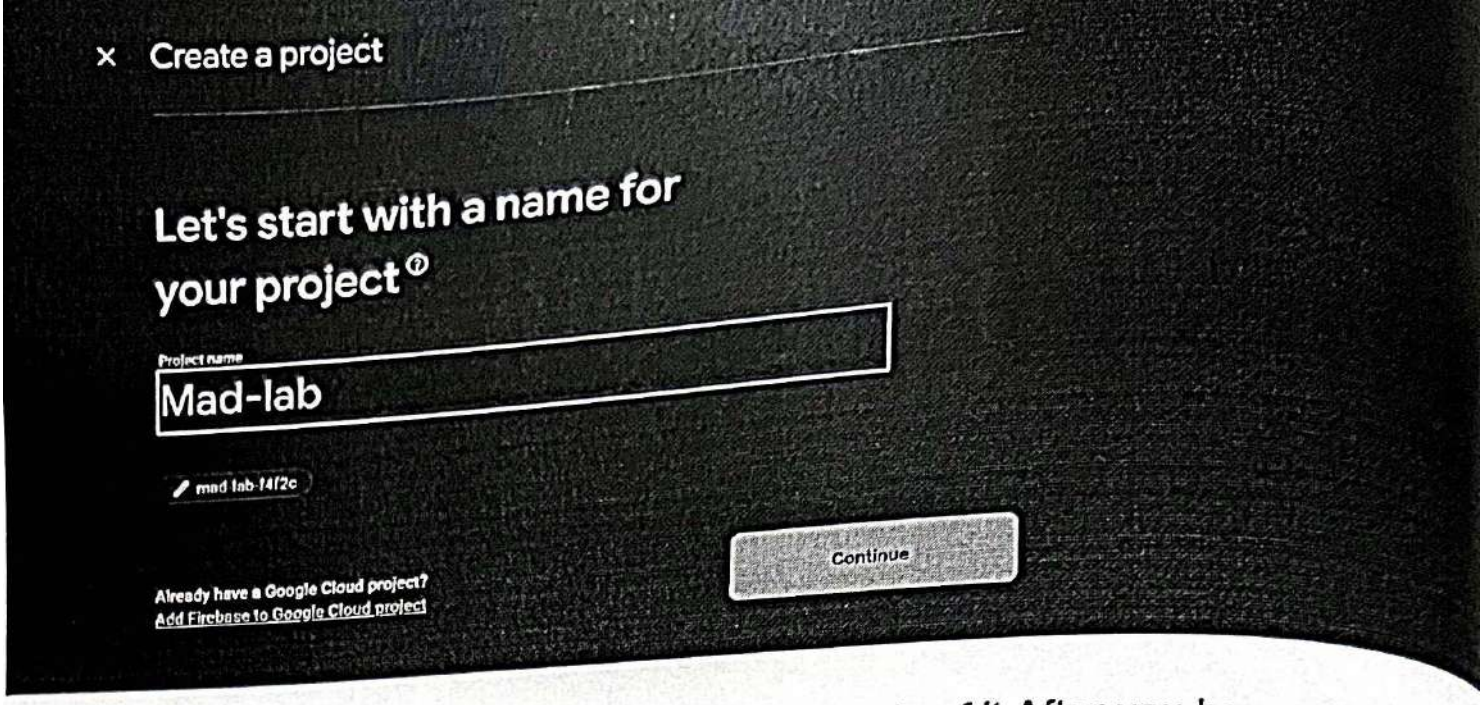
9) Test Firebase Integration:

flutter clean

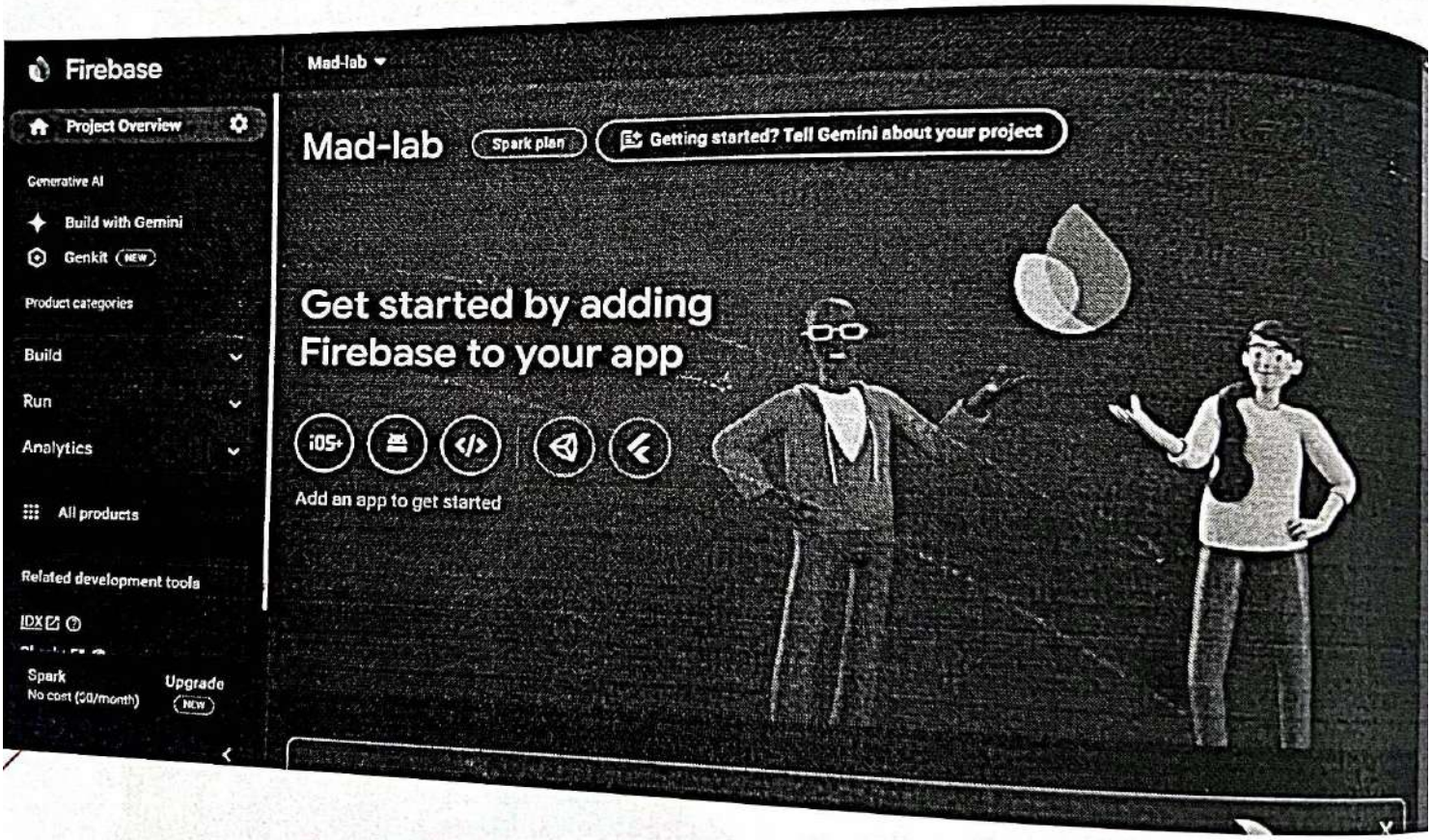
flutter pub get

flutter run

After that you will see the following new folders and files

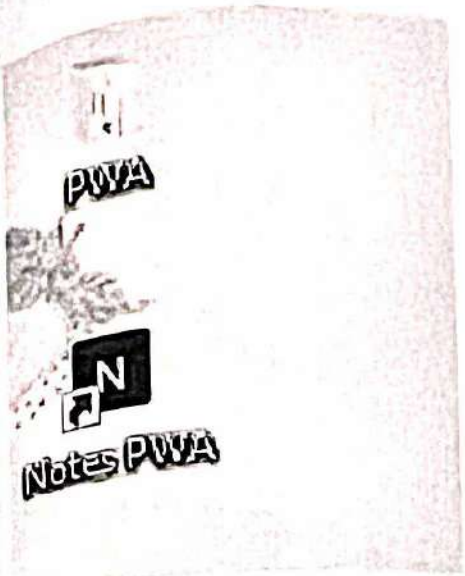


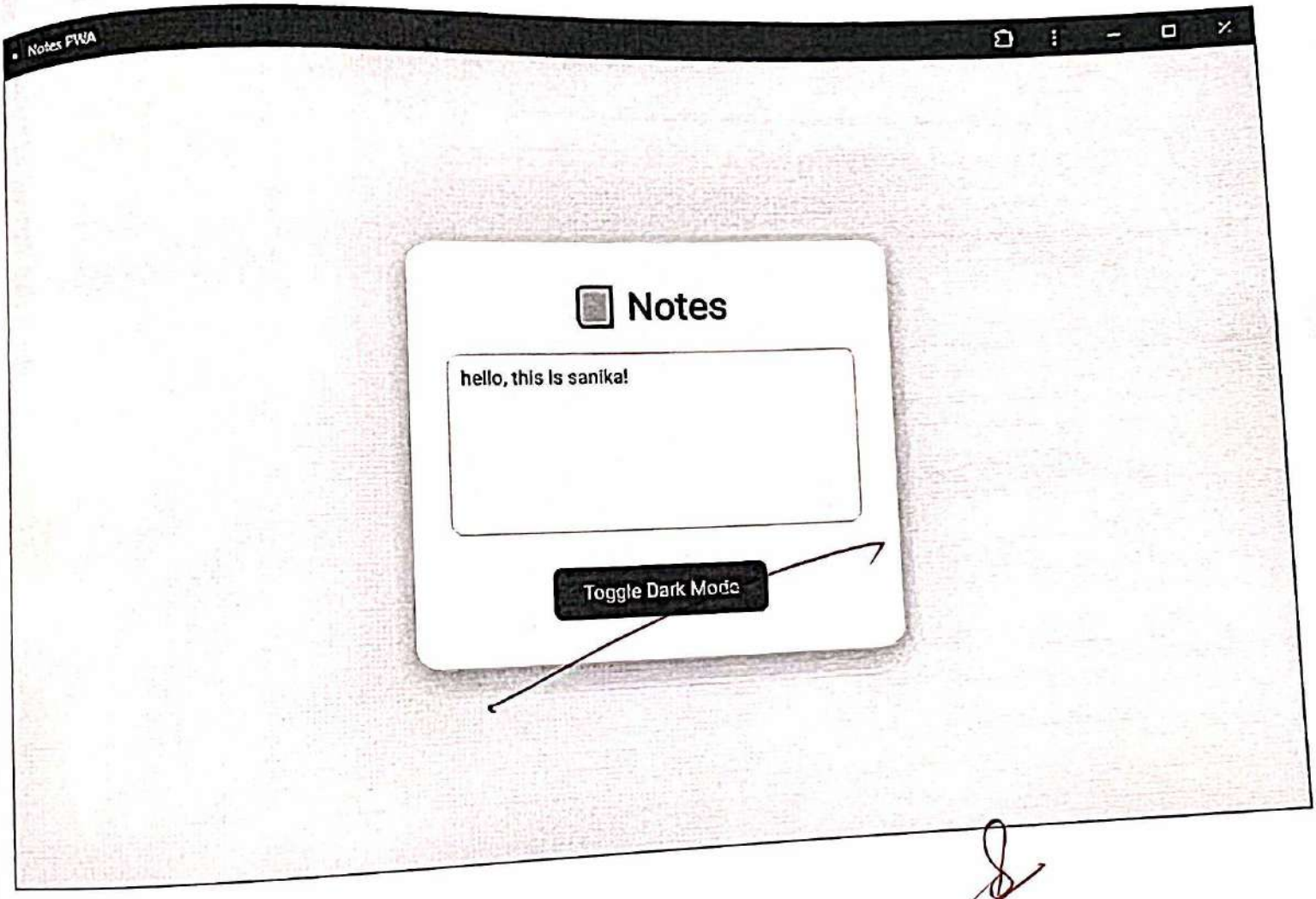
4) Select appropriate settings. Below is one example of it.After you have completed all steps you will see the following.

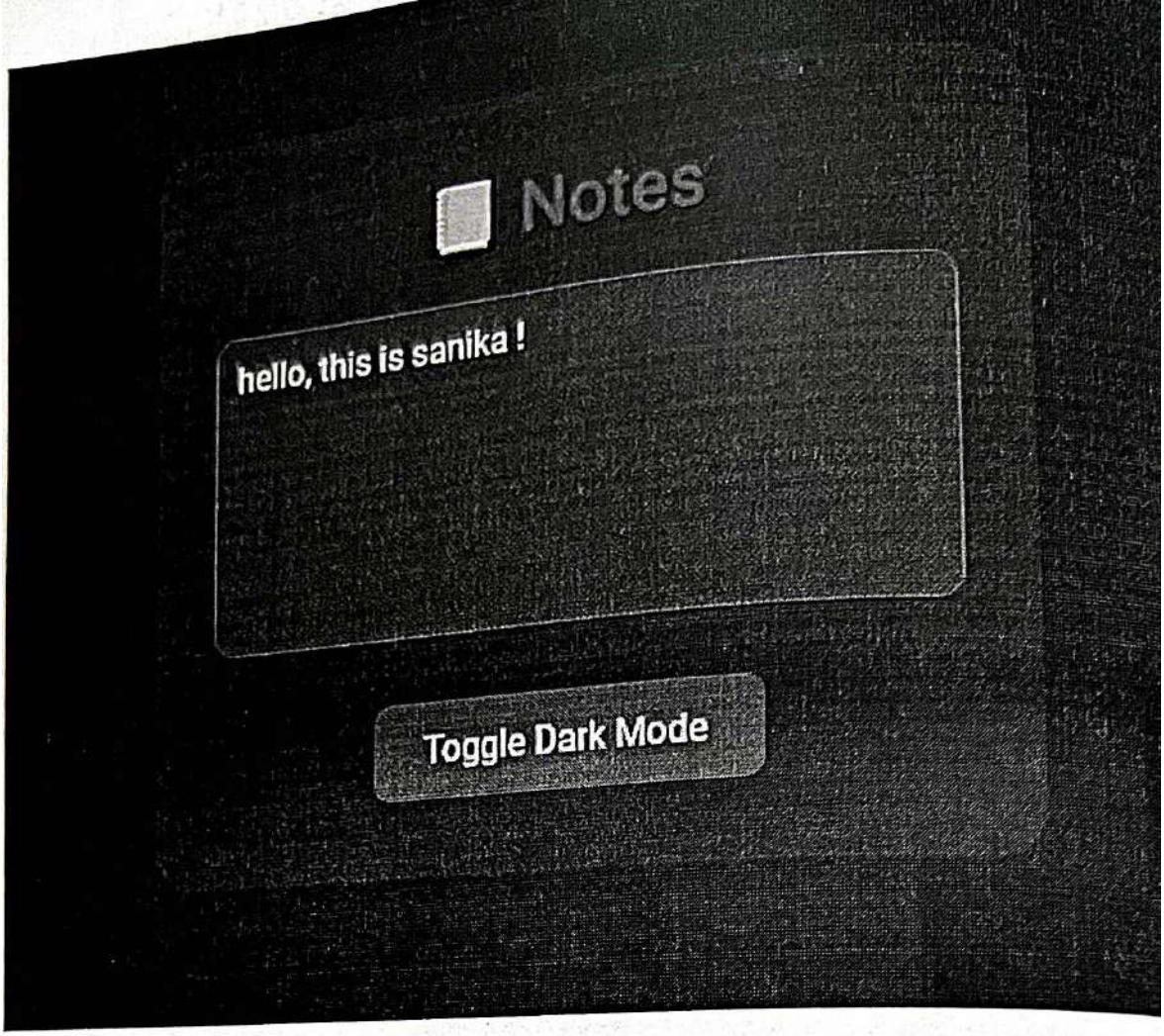


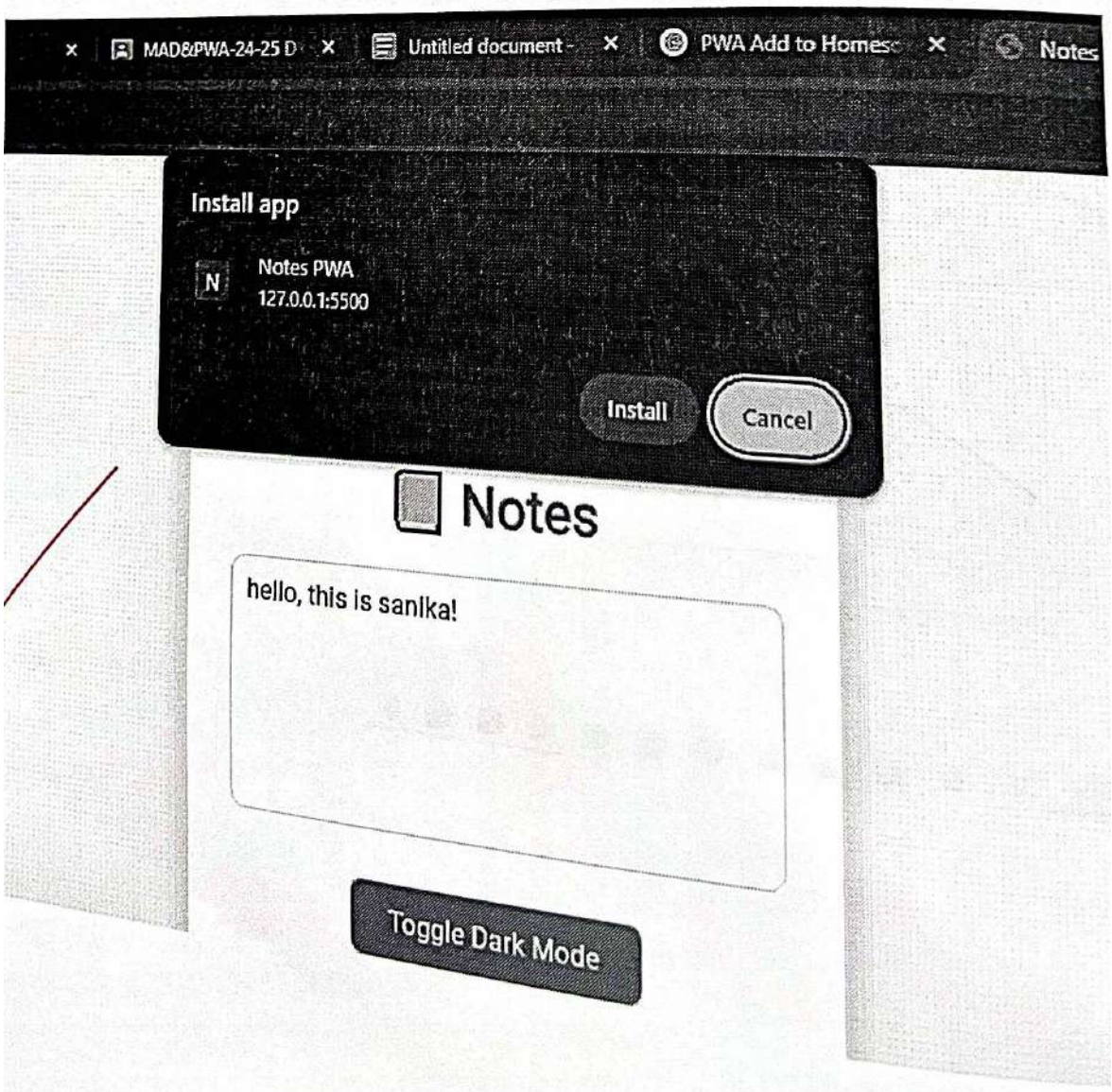
5) After these steps we need to verify-

- google-services.json (for Android) is in android/app/
- GoogleService-Info.plist (for iOS) is in ios/Runner/


PWA
N
Notes PWA


Notes PWA
Notes
hello, this is sanika!
Toggle Dark Mode


Notes
hello, this is sanika !
Toggle Dark Mode


MAD&PWA-24-25 D
Untitled document -
PWA Add to Homesc
Notes
Install app
N
Notes PWA
127.0.0.1:5500
Install
Cancel
Notes
hello, this is sanika!
Toggle Dark Mode

index.html | manifest.json × | serviceworker.js

manifest.json > ...

```
{
    "name": "Notes PWA",
    "short_name": "Notes",
    "start_url": ".",
    "display": "standalone",
    "background_color": "#f2f2f2",
    "theme_color": "#673AB7",
    "description": "A simple offline notes app with dark mode and Add to Home Screen feature.",
    "icons": [
      {
        "src": "icons/icon-192.png",
        "sizes": "192x192",
        "type": "image/png"
      },
      {
        "src": "icons/icon-512.png",
        "sizes": "512x512",
        "type": "image/png"
      }
    ]
  }
```

Notes PWA

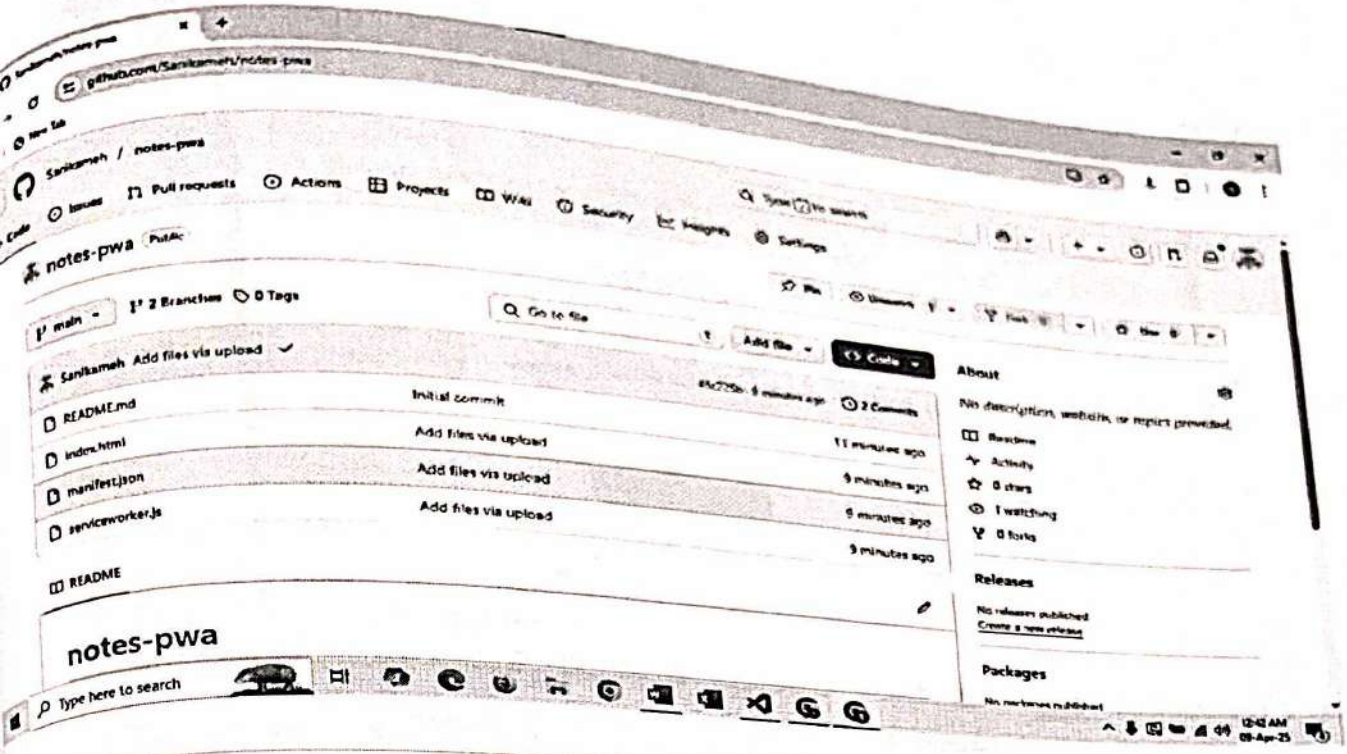
File C:/Users/DELL/Desktop/PWA/index.html

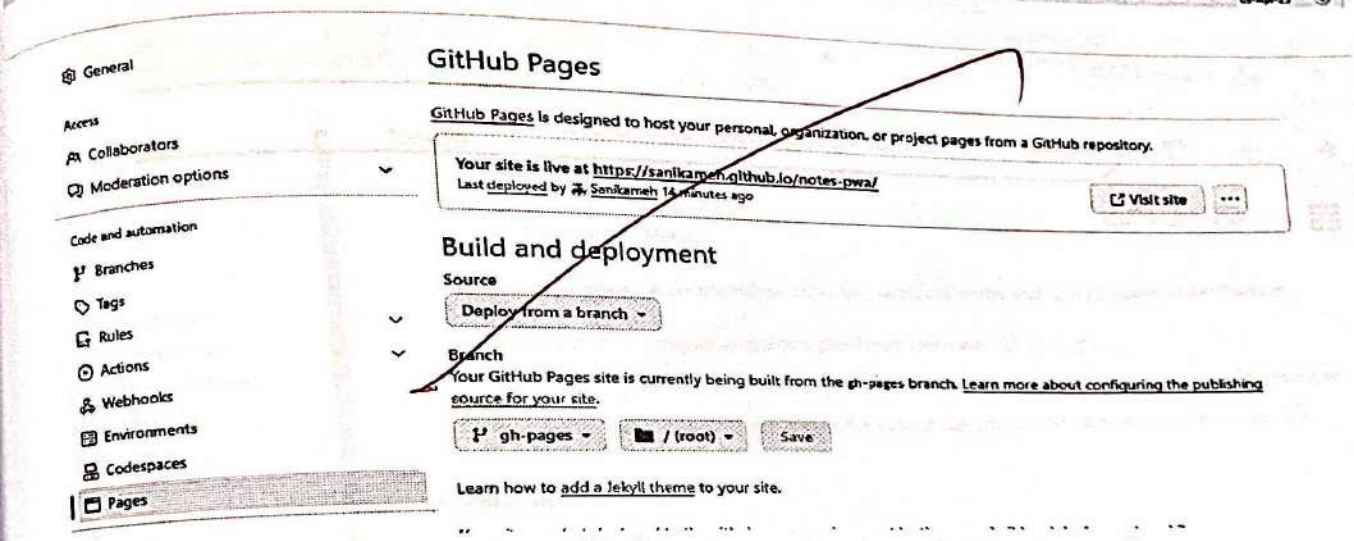
Notes

hello, this is sanika !

Toggle Dark Mode

Type here to search





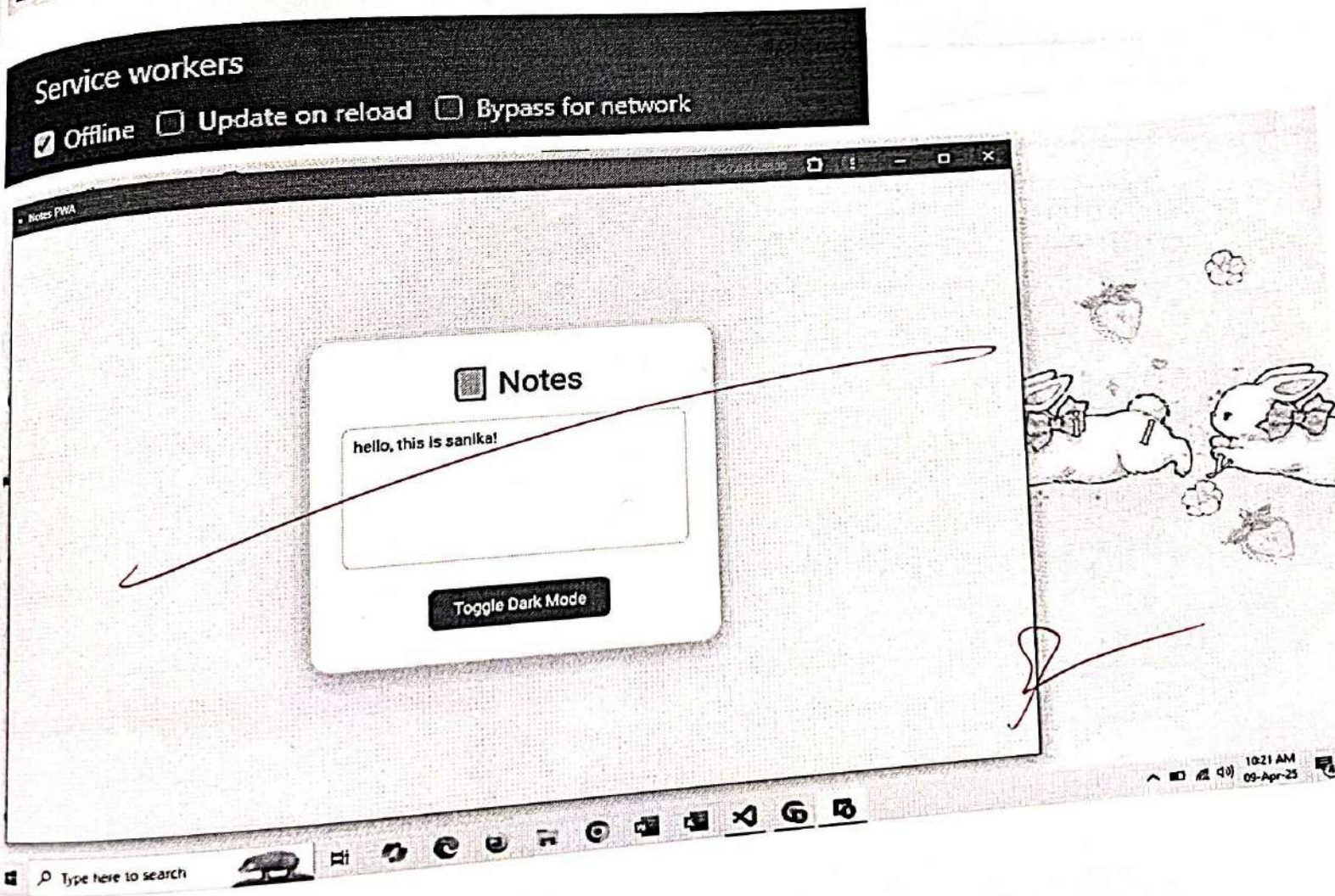
GitHub Pages

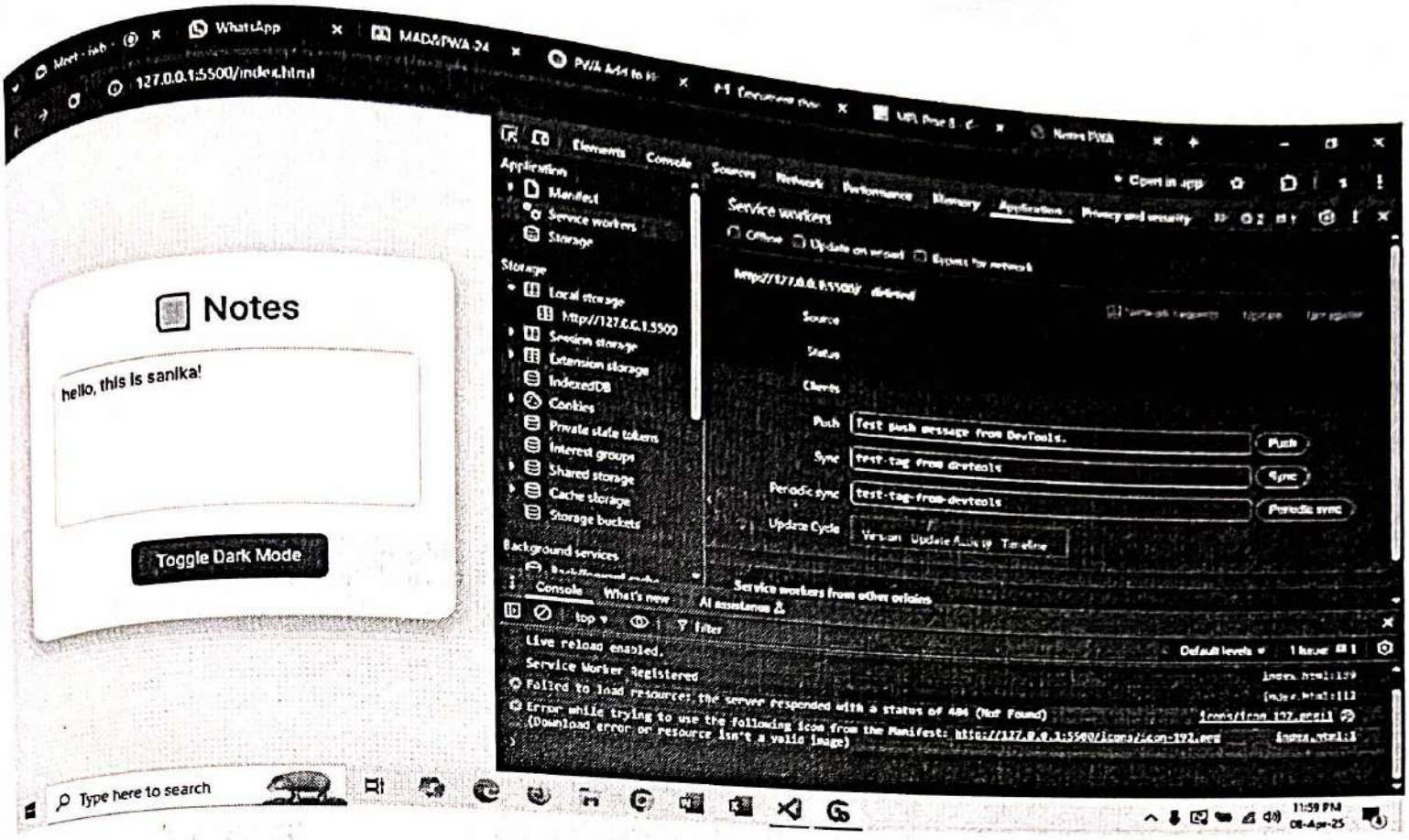
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.

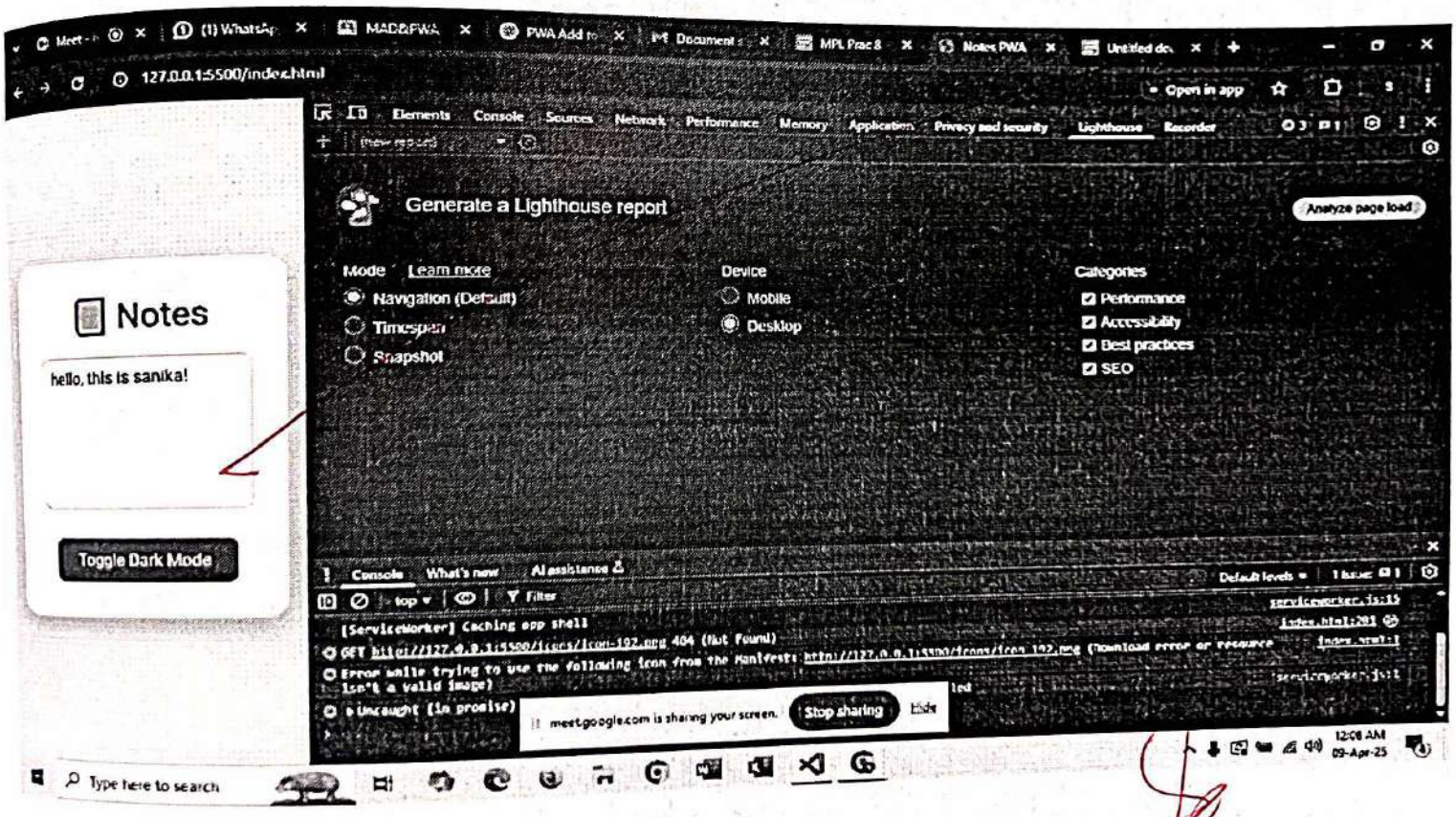
Your site is live at https://sanikameh.github.io/notes-pwa/
Last deployed by Sanikameh 14 minutes ago

Visit site ...

Notes PWA
127.0.0.1:5500/index.html
Notes
hello, this is sanika!
Toggle Dark Mode
Elements
Console
Sources
Network
Performance
Memory
Application
Privacy and security
Lighthouse
Recorder
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
Storage buckets
Background services
Back/forward cache
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5500/ - deleted
Source
Status
Clients
Push
Test push message from DevTools.
Push
Sync
test-tag-from-devtools
Sync
Periodic sync
test-tag-from-devtools
Periodic sync
Update Cycle
Versions
Update Activity
Timeline
Service workers from other origins
Console
What's new
AI assistance
What's new in DevTools 135
See all new features
new
See the highlights from Chrome 135
Type here to search
10:18 AM
09-Apr-25


Service workers
Offline
Update on reload
Bypass for network
Notes PWA
Notes
hello, this is sanika!
Toggle Dark Mode
Type here to search
10:21 AM
09-Apr-25


127.0.0.1:5500/index.html
Notes
hello, this is sanika!
Toggle Dark Mode
Elements
Console
Sources
Network
Performance
Memory
Application
Privacy and security
Lighthouse
Recorder
Manifest
Service workers
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
Storage buckets
Background services
Push
Test push message from DevTools.
Sync
test-tag from devtools
Periodic sync
test-tag-from-devtools
Update Cycle
Service workers from other origins
Live reload enabled.
Service Worker registered
Failed to load resource: the server responded with a status of 404 (Not Found)
Type here to search


127.0.0.1:5500/index.html
Notes
hello, this is sanika!
Toggle Dark Mode
Generate a Lighthouse report
Analyze page load
Mode
Learn more
Navigation (Default)
Timespan
Snapshot
Device
Mobile
Desktop
Categories
Performance
Accessibility
Best practices
SEO
[ServiceWorker] Caching app shell
GET http://127.0.0.1:5500/icons/icon-192.png 404 (Not Found)
Error while trying to use the following icon from the Manifest: http://127.0.0.1:5500/icons/icon-192.png (Download error or resource isn't a valid image)
Uncaught (in promise)
meet.google.com is sharing your screen.
Stop sharing
Hide
Type here to search

Application
- Manifest
- Service workers
- Storage

Storage
- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage
- Storage buckets

Background services
- Back/forward cache
- Background fetch
- Background sync
- Bounce tracking miti...
- Notifications
- Payment handler
- Periodic background ...
- Speculative loads

Console | AI assistance | What's new

App Manifest

manifest.json

Errors and warnings

⚠ Richer PWA Install UI won't be available on desktop. Please add at least one screenshot with the form_factor set to wide.

⚠ Richer PWA Install UI won't be available on mobile. Please add at least one screenshot for which form_factor is not set or set to a value other than wide.

⚠ Icon https://sanikameh.github.io/notes-pwa/icons/icon-192.png failed to load

⚠ Icon https://sanikameh.github.io/notes-pwa/icons/icon-512.png failed to load

⚠ Most operating systems require square icons. Please include at least one square icon in the array.

Installability

⚠ No supplied icon is at least 144 pixels square in PNG, SVG, or WebP format, with the purpose attribute unset or set to 'any'.

Identity

Name	Notes PWA
Short name	Notes
Description	A simple offline notes app with dark mode and Add to Home Screen feature.
Computed App ID	https://sanikameh.github.io/notes-pwa/ ⓘ Learn more

Application
- Manifest
- Service workers
- Storage

Storage
- Local storage
- Session storage
- Extension storage
- IndexedDB
- Cookies
- Private state tokens
- Interest groups
- Shared storage
- Cache storage
- Storage buckets

Background services
- Back/forward cache
- Background fetch
- Background sync
- Bounce tracking miti...
- Notifications
- Payment handler
- Periodic background ...
- Speculative loads

Console | AI assistance | What's new

Identity

Name	Notes PWA
Short name	Notes
Description	A simple offline notes app with dark mode and Add to Home Screen feature.
Computed App ID	https://sanikameh.github.io/notes-pwa/ ⓘ Learn more

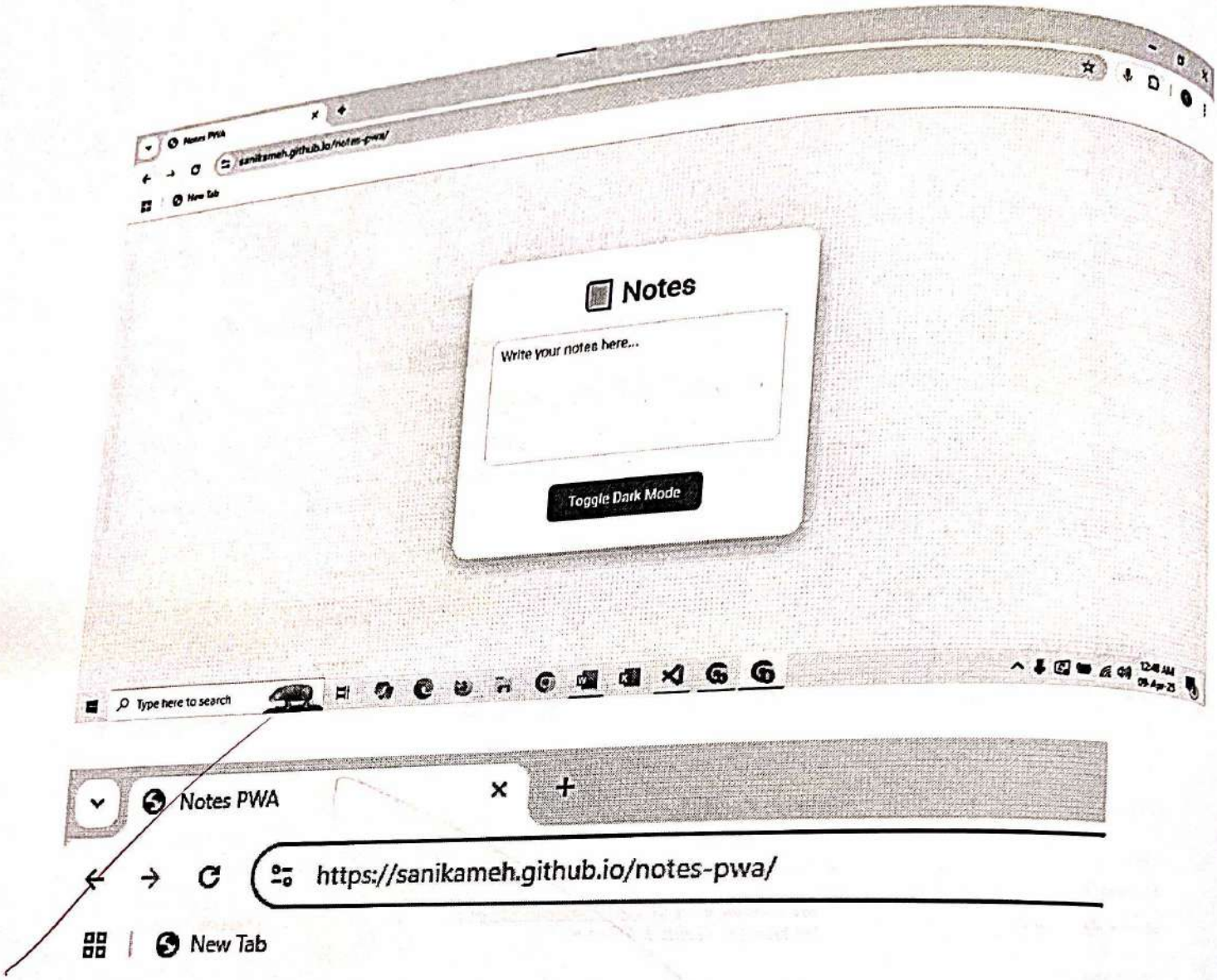
Note: `id` is not specified in the manifest, `start_url` is used instead. To specify an App ID that matches the current identity, set the `id` field to `/notes-pwa/`.

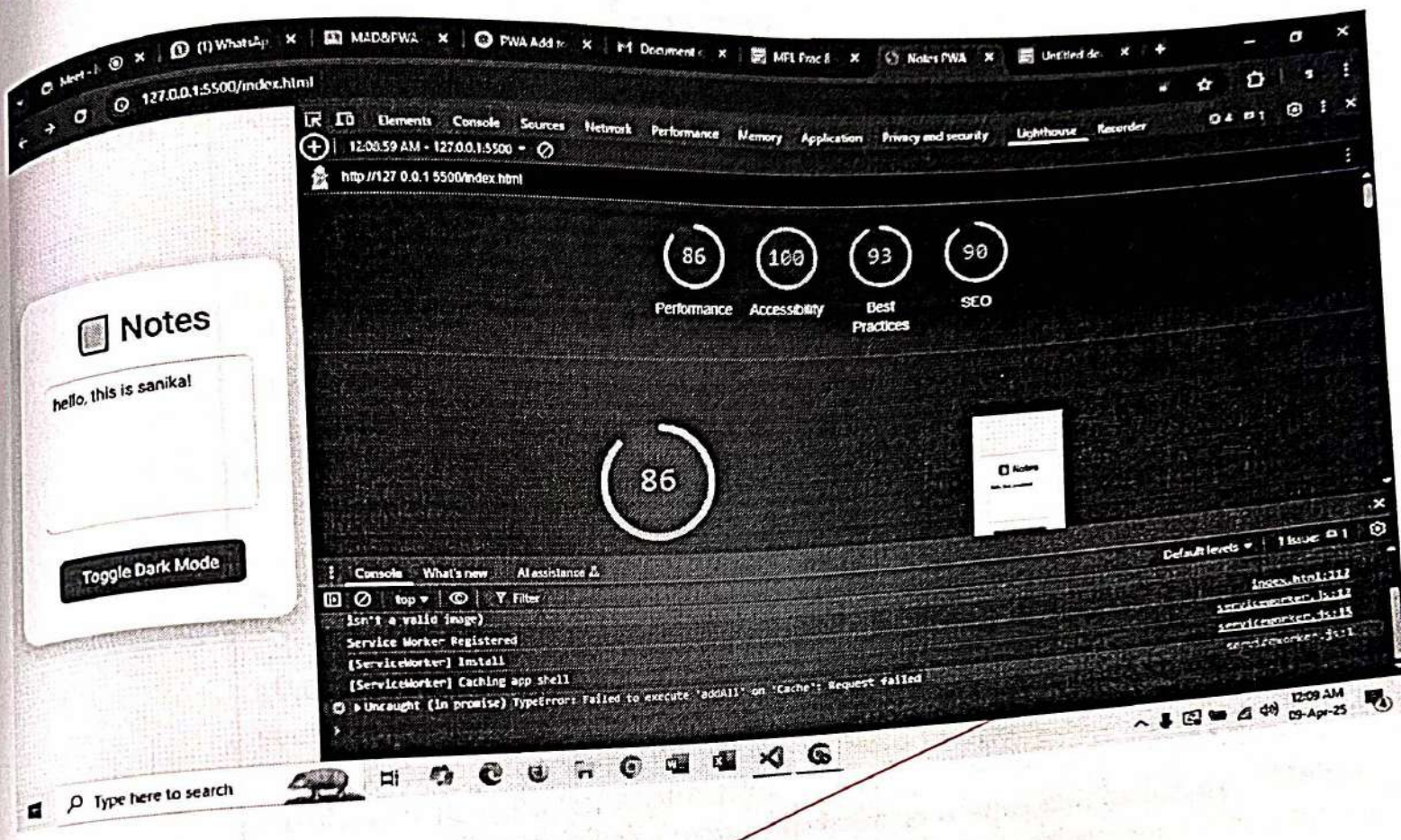
Presentation

Start URL	
Theme color	■ #673AB7
Background color	□ #f2f2f2
Orientation	
Display	standalone

Protocol Handlers

ⓘ Define protocol handlers in the manifest to register your app as a handler for custom protocols when your app is installed.


Notes PWA
sanikameh.github.io/notes-pwa/
New Tab
Notes
Write your notes here...
Toggle Dark Mode
Type here to search
Notes PWA
https://sanikameh.github.io/notes-pwa/
New Tab


127.0.0.1:5500/index.html
Elements
Console
Sources
Network
Performance
Memory
Application
Privacy and security
Lighthouse
Recorder
http://127.0.0.1:5500/index.html
86
100
93
90
Performance
Accessibility
Best Practices
SEO
86
Notes
hello, this is sanika!
Toggle Dark Mode
Console
What's new
AI assistance
Service Worker Registered
[ServiceWorker] Install
[ServiceWorker] Caching app shell
Uncaught (in promise) TypeError: Failed to execute 'addAll' on 'Cache': Request failed
Type here to search

Sanika Mehetre
D15B - 34

MPL Assignment - 1 (A+) (05)

Q1. a) Explain the key features and advantages of using flutter for mobile app development.

⇒ 1) Single codebase for multiple platforms
Write one codebase for both Android and iOS, reducing development effort and maintenance.

2) Hot Reload : Instantly see changes in the app without restarting, making development faster and more interactive

3) Fast Performance : Uses the Dart language and a compiled approach for smooth and high performance app.

4) Open source & strong community support:
Backed by Google and a large developer community, ensuring continuous improvements and resources.

Advantages:

i) Faster Development Time : Hot Reload and single codebase reduce development time significantly.

ii) Cost Effective : Since the same code runs on both Android and iOS, businesses save on development and maintenance cost.

iii) Reduced performance issues : The app runs natively without relying on intermediate bridges like in react native reducing lag

Sundaram
FOR EDUCATIONAL USE

b) Discuss how the flutter framework differs from traditional approaches and why it has gained popularity in the development community.

⇒ 1.] Single codebase V/S separate codebase:

- Traditional Approach: Developers need to write separate code for Android (Java/Kotlin) and IOS (Swift)
- Flutter: Uses a single Dart based codebase for both platforms reducing development time and effort.

2.] Rendering engine V/S Native UI components:

- Traditional Approach: Relies on platform, native UI components which can lead to inconsistencies and performance issues.
- Flutter: Uses the skia rendering engine to draw everything from scratch ensuring a consistent UI across devices.

Why Flutter has gained popularity

1) Faster development with hot reload:

Developers can instantly see UI changes without restarting the app making the iteration process much quicker.

2) cross platform efficiency: Businesses save time and resources by maintaining a single codebase for multiple platforms.
3) consistent UI across devices: Since flutter does not rely on native components, the UI looks and behaves the same across different OS versions.
4) Improved performance: AOT compilation and direct accesses to GPO rendering ensure smooth animations and high performance
5) Easy Integration with Backend Technologies: Works well with Firebase, REST API, GraphQL and other backend technologies.

Q2.a) Describe the concept of the widget tree in flutter. Explain how widget composition is used to build complex user interfaces.

→ Widget tree in flutter:

In flutter, the widget tree is the fundamental structure that represented the UI of an application. It is a hierarchical arrangement of widgets where each widget defines a part of the user interface.

Widget composition in flutter:

Widget composition refers to building complex UIs by combining smaller, reusable widgets. Instead of creating large,

```
ElevatedButton(
  onPressed : (){
    print("Button Pressed");
    },
    child : Text("Submit"),
    ),
  },
  );
```

3) Display & Styling Widgets

- Text - displays text on the screen
- Image - shows image from assets network or memory
- Icon - Displays icon.
- Card - A material design card with rounded corners and elevations.

eg:
```
Column(
  children : [
    Text("Welcome to Flutter!", style : TextStyle
        (Fontsize : 24, Fontweight : FontWeight.bold)),
    Image.network("https://flutter.dev/images/
                  flutter_logo_sharing.png"),
  ],
  );
```

Q3. a) Discuss the important state of management in flutter applications.

⇒ In Flutter, state refers to data that can change during the lifetime of an application.

for building the UI

- Material App: The root widgets of a flutter app that provides essential configurations.
- Scaffold: Provides a basic layout structure, including an app bar, body floating action button etc.
- Container: A versatile widget used for styling, padding, margin and background customization.

```
eg: MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Flutter Widget Tree")),
        body: Container(
          padding: EdgeInsets.all(16.0),
          child: Text("Hello, Flutter!"),
        ),
      ),
    );
```

2) Input & Interaction Widgets:

TextField - Accepts text input from users.

ElevationButton - A button with elevation.

GestureDetector - Detects gestures like taps, swipes and long presses

```
eg: Column(
      children: [
        TextField(decoration: InputDecoration(labelText:
                                "Enter name"
```

monolithic UI components.

Example :

```
class Profilecard extends statelessWidget {
    final string name;
    final string imageUrl;
  Profilecard ({ required this.name, required this.imageUrl})
    @ override
    widget build (BuildContext context) {
        return Card (
            child. Column (
                children. [
                    Image. network (imageurl)
                    SizedBox (height : 10)
                    Text (name, style: Textstyle (fontsize: 20,
                                fontweight: Fontweight. bold)
                ]
            )) }
```

Benefits of Widgets composition :

1) Reusability : small widgets can be reused in different parts of the app.
2) Maintainability : Breaking UI into smaller widgets makes it easier to debug and update.

Q2. b) Provide examples of commonly used widgets and their roles in creating a widget tree.

→ 1) Structural widgets :

These widgets act as the foundation

Step 1: Create a firebase project
- Go to firebase console
- Click on "Add project" and enter a project name
- Configure google analytics if needed. Then click create.

Step 2: Register the flutter app with Firebase
- In the firebase project dashboard click "Add app" and select Android or IOS based on your platform.
- For Android: Enter the android package name and download the google services, json file and place it in android/appl.
- For IOS
  Enter the IOS Bundle identifier
  Download the Google Service - info.

Step 3: Install firebase dependencies
Add firebase dependencies in nullpro.yaml
firebase core
firebase auth
cloud firestore
Run: flutter pub get

Provider - App wide state
Pros: Lightweight, recommended by flutter, efficient.
Cons: Boilerplate code for nested providers
Best use case: Medium Scale apps (eg: authentication, themes, API data)

Riverpod App - Wide state (more scalable than provider)
Pros: Eliminates Providers limitations, improves performance
Cons: Requires learning new concepts
Best use case: Large apps needing global state (eg: shopping cart, user sessions)

Q4.a) Explain the process of integrating Firebase with a flutter application. Discuss the benefits of using Firebase as a backend solution.

⇒ Firebase provides a powerful backend solution for flutter applications offering services like authentication, real time databases, cloud functions, storage and more.

Steps to integrate Firebase with flutter

This includes:
- User input
- UI changes
- Network changes
- Animation states

There are two types of states:
1) Ephemeral state: Small, UI specific state that doesn't affect the whole app.
2) App wide status: Data shared across multiple widgets

Importance of State Management:
- Effecient UI updates: Flutter's UI is rebuilt whenever state changes
- Code Maintainability & Scalability: Managing state properly makes the code modular, readable and scalable for larger applications

Q3.b) Compare and contrast the different state management approaches available in flutter, such as setstate, Provider and Riverpod. Provide scenarios where each approach is suitable.

→ Setstate - Local state

Pros - Simple built in easy to use

Cons - Not scalable causes unnecessary re-renders

Best use cases - small UI updates

Sundaram

4) Firebase Cloud Messaging (FCM)
Enables push notifications and messaging between users
eg: FirebaseMessaging.Instance.SubscribeToTopic("news");

5) Firebase Analytics:
Tracks user interaction and app performance.
eg: FirebaseAnalytics.analytics.FirebaseAnalytics.instance;
analytics.logEvent(name: "button clicked", parameters: {"button", "subscribe"});

6) Firebase Hosting:
Deploys and serves web applications securely with automatic SSL.

Data synchronization in Firebase
Firebase ensures real-time data synchronization across multiple devices and platforms using Firebase and realtime Database.
1) Cloud Firestore Sync Mechanism:
Uses realtime listeners to update UI instantly when data changes
2) Realtime Database Sync Mechanism:
Uses persistent websocket connections for live updates.
3) Offline Data Sync:
Firestore caches data locally and syncs chang

FOR EDUCATIONAL USE
Sundaram

Q4.b) Highlight the firebase services commonly used in flutter development and provide a brief overview of how data synchronization is achieved.

⇒ Firebase provides a suite of backend services that simplify flutter app development.

1) Firebase authentication:
Enables secure authentication using email/ password, phone number and third party providers like google, facebook.

2) Cloud Firebase:
Stores and syncs data in real time across devices. Supports structured data, queries and offline access.

eg: Firebase.Firestore.instance.collection('users')
.add({
'name': 'John Doe',
'email': 'johndoe@example.com',
});

3) Realtime Database:
A realtime, json-based database that automatically updates data across devices.

eg: DatabaseReference ref = FirebaseDatabase.instance.
ref("message");
ref.set({"text": "Hello, Firebase!"});

Step 4: Configure Firebase for Android & iOS

For Android:

1) Open android / build.gradle and ensure the following classpath 'com.google-services:4.3.10'
2) Open android/app/build.gradle and add at the bottom apply plugin: 'com.google.gms.google services'

Step 5: Initialize Firebase in flutter

```
void main() async {
  widgetFlutterBinding.ensureInitialized();
  await FirebaseInitializeApp();
    runApp(MyApp());
```

Benefits of using Firebase:

1) Easy to setup & scale: No need to manage backend infrastructure.
2) Authentication: No need to manage Provides email, password.
3) Cloud Storage: Secure file storage for images, videos and documents.
4) Push Notifications: Send real-time notifications to user across different platforms.

Sanika Menebre
D15B - 34

05/05

MPL Assignment - 2

Q1. Define Progressive Web App (PWA) and explain its significance in modern web development. Discuss the key characteristics that differentiate PWAs from traditional mobile apps.

⇒ 1). A Progressive Web App (PWA) is a type of web application that combines the best features of web and mobile apps, offering a fast, reliable, and engaging user experience.

2) PWAs use modern web capabilities to deliver and app-like experience while being accessible through a web browser.

Significance in Modern Web Development

- Cross-platform compatibility : PWAs work on any device with a modern web browser, reducing the need for separate development for IOS and Android.

- Offline functionality : Thanks to Service Workers, PWAs can work even without an internet connection.

- App-like experience : PWAs provide a full-screen experience, push-notifications and home screen installation, making them

similar to native apps.

- SEO-friendly: Unlike traditional mobile apps, PWAs are indexable by search engines, improving discoverability.

PWA	Traditional Mobile App
1) No app store needed; installed via browser	Installed via App store or Play store
2) Updates are automatic, no user intervention needed.	Updates are required to be manual from the store.
3) Fast loading with caching	Faster but requires most storage.
4) Offline access with service workers	Offline access only if designed so.
5) Development cost is lower due to single codebase	Development cost is higher due to separate development for platforms.

Sundaram

Q2. Define responsive web design and explain its importance in the context of Progressive Web Apps. Compare and contrast responsive, fluid and adaptive web design approaches.

⇒ 1) Responsive Web Design (RWD) ensures that web applications automatically adjust their layout, images, and content based on the screen size and resolution of the user's device.

2) This approach is crucial for PWAs, as they must provide a seamless experience across various devices, from desktops to smartphones.

Importance of Responsive Web Design in PWAs

- Ensures consistency in user experience across different screen sizes.
- Enhances usability by adapting navigation and content layout.
- Reduces development effort by eliminating the need for multiple versions of the same app.

1.] Responsive:

1) Uses flexible grids, media queries, and scalable images to adjust layout based on screen size.

2) It provides a seamless experience across all devices but can be complex to implement.

Sundaram
FOR EDUCATIONAL USE

2.] Fluid :

1) Uses percentage - based layouts, making elements scale relative to the screen size
2) It has smooth resizing without fixed breakpoints but it may lead to poor readability if not carefully designed.

3.] Adaptive :

1) Uses fixed layouts for specific screen sizes, loading a different design depending on the device.
2) It has optimized experience for each device type but it requires multiple versions, increasing the maintenance effort.

Q3. Describe the lifecycle of Service Workers, including registration, installation and activation phases.

⇒ 1) A Service Worker is a script that runs in the background, enabling features like caching and push notifications.
2) Its lifecycle consists of three main phases.

1.] Registration :

The Service Worker is registered using Javascript in the main web page.
The browser checks if a new Service Worker

4) It handles large and complex data efficiently.
5) It supports transactions and indexing for better performance.

```
const dbPromise = idb.open('my-database', 1, upgradeDB=>{
  if (!upgradeDB.objectStoreNames.contains('items')) {
    upgradeDB.createObjectStore('items', {keyPath:'id'});

// Storing data
function saveData(data) {
  dbPromise.then(db => {
    const tx = db.transaction('items', 'readwrite');
    const store = tx.objectStore('items');
    store.put(data);
    return tx.complete;
  });
}

// Retrieving data
function getData(id) {
  return db.Promise.then(db => {
    Const tx = db.transaction('items', 'readonly');
    const store = tx.objectStore('items');
  });
}
```

This approach allows PWAs to maintain functionality even when offline by caching essential data and retrieving it when needed.

FOR EDUCATIONAL USE

3.] Activation:

The old Service Worker is replaced with the new one. Cleanup of outdated caches is performed.

```
self.addEventListener('activate', event => {
  event.waitUntil(
    caches.keys().then(keys => {
      return Promise.all(
        keys.filter(key => key !== 'app-cache').
      );
    })
  );
});
```

After activation, the Service Worker starts controlling the app and can handle fetch requests

Q4. Explain the use of IndexedDB in the Service Worker for data storage.

→ 1) IndexedDB is a low-level, NoSQL database that allows web applications to store large amounts of structured data.

2) Unlike localStorage, it provides effecient read/write operations and works asynchronously.

3) It enables offline data storage and retrieval.

file is available.

```
If ('serviceWorker' in navigator) {
  navigator.serviceWorker.register('/service-worker.js')
    .then(() => console.log('Service Worker registered'))
    .catch(err => console.log('Registration failed, err));
}
```

2.] Installation:

The Service Worker is downloaded and installed. This phase is used for caching essential assets.

```
self.addEventListener('install', event => {
  event.waitUntil(
    caches.open('app-cache').then(cache => {
      return cache.addAll([
        '/'
        '/index.html',
        '/styles.css',
        '/script.js'
      ]);
    })
  );
});
```