Aaryan Kalbhar
D15B - 26

MPL Project - Plan 2 Shop

Problem Statement: Many individuals struggle with meal planning and grocery shopping due to inefficiencies in existing apps, which lack seamless integration and personalized grocery list and effective food waste management. This leads to difficulties in maintaining a consistent and healthy meal routine while minimizing effort and costs.

Description: Plan2Shop is a smart meal planning and grocery list app designed to simplify meal preparation and shopping. It helps users effortlessly plan weekly meals, generate personalised grocery lists, reduce food waste and maintain a healthy routine, with suggestions.

Features:
1) Dynamic Grocery List - Auto-generated based on selected meals categorised by sections
2) Custom Grocery List - Manually created by users
3) Meal Prep List - Organises ingredients and steps for cooking and meal prep.
4) Reminders - Notify about shopping and meal prep time.

Reference App - Mealime.

Sundaram
FOR EDUCATIONAL USE

Name : Aaryan Kalbhor
Div : D15B
Roll No : 26

EXP 1: Installation and Configuration of Flutter Environment.

Aim: To install and configure the Flutter Environment, also install Android Studio.

Install the Flutter SDK

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

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

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

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

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

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

Step 4.3: In the above window, click on New->write path of Flutter bin folder in variable value
Step 5: Now, run the $ flutter command in command prompt.

```
Command Prompt - flutter

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

C:\Users\Aaryan>flutter
Manage your Flutter app development.

Common commands:

  flutter create <output directory>
    Create a new Flutter project in the specified directory.

  flutter run [options]
    Run your Flutter application on an attached device or in an emulator.

Usage: flutter <command> [arguments]

Global options:
-h, --help                  Print this usage information.
-v, --verbose               Noisy logging, including all shell commands executed.
                            If used with "--help", shows hidden options. If used with "flutter doctor", shows additional
                            diagnostic information. (Use "-vv" to force verbose logging in those cases.)
-d, --device-id             Target device id or name (prefixes allowed).
    --version               Reports the version of this tool.
    --enable-analytics      Enable telemetry reporting each time a flutter or dart command runs.
    --disable-analytics     Disable telemetry reporting each time a flutter or dart command runs, until it is
                            re-enabled.
    --suppress-analytics    Suppress analytics reporting for the current CLI invocation.

Available commands:
```

Step 6: Now, run the $ flutter doctor command. This command checks for all the requirements of Flutter app development and displays a report of the status of your Flutter installation., you will find the details of all missing tools, which required to
run Flutter as well as the development tools that are available but not connected with the device.

```
Command Prompt - flutter d

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

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

[✓] Chrome - develop for the web
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[!] Android Studio (not installed)
[✓] IntelliJ IDEA Community Edition (version 2024.3)
[✓] VS Code (version 1.96.4)
[✓] Connected device (3 available)
[✓] Network resources

! Doctor found issues in 3 categories.

C:\Users\Aaryan>
```

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

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

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

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

Step 7.4: In the above screen, click Next-> Finish.

Step 7.5: run the $ flutter doctor command and Run flutter doctor --android-licenses command.

```
Command Prompt

nment or any agency thereof or the foreign government from where it is shipping requires an export license, or other gov
ernmental approval, without first obtaining such license or approval. Recipient also agrees to implement measures to ens
ure that foreign national employees are authorized to receive any information controlled by U.S. export control laws. An
 export is "deemed" to take place when information is released to a foreign national wherever located.

10.7 Special Terms for Pre-Release Materials. If so indicated in the description of the Evaluation Software, the Evaluat
ion Software may contain Pre-Release Materials. Recipient hereby understands, acknowledges and agrees that: (i) Pre-Rele
ase Materials may not be fully tested and may contain bugs or errors; (ii) Pre-Release materials are not suitable for co
mmercial release in their current state; (iii) regulatory approvals for Pre-Release Materials (such as UL or FCC) have n
ot been obtained, and Pre-Release Materials may therefore not be certified for use in certain countries or environments
or may not be suitable for certain applications and (iv) MIPS can provide no assurance that it will ever produce or make
 generally available a production version of the Pre-Release Materials . MIPS is not under any obligation to develop and
/or release or offer for sale or license a final product based upon the Pre-Release Materials and may unilaterally elect
 to abandon the Pre-Release Materials or any such development platform at any time and without any obligation or liabili
ty whatsoever to Recipient or any other person.

ANY PRE-RELEASE MATERIALS ARE NON-QUALIFIED AND, AS SUCH, ARE PROVIDED ?AS IS? AND ?AS AVAILABLE?, POSSIBLY WITH FAULTS,
 AND WITHOUT REPRESENTATION OR WARRANTY OF ANY KIND.

10.8 Open Source Software. In the event Open Source software is included with Evaluation Software, such Open Source soft
ware is licensed pursuant to the applicable Open Source software license agreement identified in the Open Source softwar
e comments in the applicable source code file(s) and/or file header as indicated in the Evaluation Software. Additional
detail may be available (where applicable) in the accompanying on-line documentation. With respect to the Open Source so
ftware, nothing in this Agreement limits any rights under, or grants rights that supersede, the terms of any applicable
Open Source software license agreement.
---------------------------------------
Accept? (y/N): y
All SDK package licenses accepted

C:\Users\Aaryan>
```

```
C:\Users\Aaryan>sdkmanager --version
C:\Users\Aaryan>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.27.3, on Microsoft Windows [Version 10.0.22631.4751], locale en-ID)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[✓] Android toolchain - develop for Android devices (Android SDK version 35.0.1)
[✓] Chrome - develop for the web
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[✓] Android Studio (version 2024.2)
[✓] IntelliJ IDEA Community Edition (version 2024.3)
[✓] VS Code (version 1.96.4)
[✓] Connected device (3 available)
[✓] Network resources

! Doctor found issues in 1 category.

C:\Users\Aaryan>
```

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

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

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

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

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

Step 9: Now, install Flutter and Dart plugin for building Flutter application in Android Studio.

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

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

Step 9.3: Restart the Android Studio.

<u>Conclusion</u>: The lab practical successfully showcased the installation and configuration of Flutter with Android Studio. The setup facilitated smooth cross-platform app development, emphasizing the significance of proper SDK setup, emulator usage, and plugin integration for efficient Flutter development.

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

void main() {
  runApp(Plan2ShopApp());
}

class Plan2ShopApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Plan2Shop',
      debugShowCheckedModeBanner: false,
      theme: ThemeData(
        primarySwatch: Colors.teal,
        // Set your custom font family defined in pubspec.yaml
        fontFamily: 'CustomFont',
      ),
      home: HomeScreen(),
    );
  }
}

class HomeScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Plan2Shop', style: TextStyle(fontFamily: 'CustomFont')),
        centerTitle: true,
      ),
      body: Column(
        children: [
          // Banner Image
          Container(
            width: double.infinity,
            height: 200,
            child: Image.asset(
              'assets/images/banner.jpg',
              fit: BoxFit.cover,
            ),
          ),
          // Search Field
          Padding(
```

```
          padding: const EdgeInsets.all(8.0),
          child: TextField(
            decoration: InputDecoration(
              labelText: 'Search Recipes',
              border: OutlineInputBorder(),
              prefixIcon: Icon(Icons.search),
            ),
          ),
        ),
        // Meal Categories List
        Expanded(
          child: ListView(
            children: [
              MealCard(title: "Breakfast", icon: FontAwesomeIcons.coffee),
              MealCard(title: "Lunch", icon: FontAwesomeIcons.utensils),
              MealCard(title: "Dinner", icon: FontAwesomeIcons.hamburger),
            ],
          ),
        ),
      ],
    ),
    // Bottom Navigation Bar with icons
    bottomNavigationBar: BottomNavigationBar(
      items: [
        BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),
        BottomNavigationBarItem(
            icon: Icon(FontAwesomeIcons.shoppingCart), label: 'Grocery List'),
        BottomNavigationBarItem(icon: Icon(Icons.settings), label: 'Settings'),
      ],
    ),
  );
 }
}

class MealCard extends StatelessWidget {
 final String title;
 final IconData icon;

 MealCard({required this.title, required this.icon});

 @override
 Widget build(BuildContext context) {
  return Card(
    margin: EdgeInsets.all(10),
```

```
      child: ListTile(
        leading: Icon(icon, size: 40, color: Colors.teal),
        title: Text(
          title,
          style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
        ),
        trailing: Icon(Icons.arrow_forward),
        onTap: () {
          Navigator.push(
            context,
            MaterialPageRoute(
                builder: (context) => MealDetailScreen(mealType: title)),
          );
        },
      ),
    );
  }
}

class MealDetailScreen extends StatelessWidget {
  final String mealType;

  MealDetailScreen({required this.mealType});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("$mealType Recipes"),
      ),
      body: GridView.count(
        crossAxisCount: 2,
        padding: EdgeInsets.all(10),
        crossAxisSpacing: 10,
        mainAxisSpacing: 10,
        children: List.generate(6, (index) {
          return Card(
            elevation: 3,
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                // Recipe Image Placeholder
                Container(
                  width: 80,
```

```
                height: 80,
                child: Image.asset(
                  'assets/images/recipe_placeholder.png',
                  fit: BoxFit.cover,
                ),
              ),
              SizedBox(height: 8),
              Text(
                "Recipe ${index + 1}",
                style: TextStyle(fontWeight: FontWeight.bold),
              ),
            ],
          ),
        );
      }),
    ),
  );
}
}
```

12:08
Plan2Shop
Unable to load asset: "assets/images/banner.jpg".
Exception: Asset not found
Search Recipes
Breakfast
Lunch
Dinner
Home
Grocery List
Settings

12:09
Breakfast Recipes
Recipe 1
Recipe 2
Recipe 3
Recipe 4
Recipe 5
Recipe 6

11:39
Login
Email
Password
Login
Don't have an account? Sign Up

11:59
Sign Up
Full Name
Email
Password
Sign Up
1:1

11:39
PLANzShop
@aaryan
Search Recipes
Breakfast
Lunch
Dinner
Home
Grocery List
Settings

11:39
Breakfast Recipes
Recipe 1
Recipe 2
Recipe 3
Recipe 4
Recipe 5
Recipe 6

11:39
Login
Email
Password
Login
Don't have an account? Sign Up

11:59
Sign Up
Full Name
Email
Password
Sign Up
1:1

23:22
52%
Recipes
Recommended Recipes
Red Sauce Pasta
Egg Masala
Chicken Steak
Avocado sandwich
Pizza
Pepperoni
Recipes
Grocery List
Settings

23:22
52%
Grocery List
Dairy
Butter
Meat
Chicken-Breast
Veggies
Bell-Peppers
Carrots
Tomatoes
Recipes
Grocery List
Settings

23:22
52%
Settings
Set Reminder
About
Logout
Recipes
Grocery List
Settings

Code:

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

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

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

class GestureExample extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text('Gesture Example')),
      body: Center(
        child: GestureDetector(
          onTap: () {
            print("Tapped!");
          },
          onDoubleTap: () {
            print("Double Tapped!");
          },
          onLongPress: () {
            print("Long Pressed!");
          },
          onPanUpdate: (details) {
            print("Swiped! ${details.delta}");
          },
          child: Container(
            padding: EdgeInsets.all(20),
            color: Colors.blue,
            child: Text(
              "Tap, Double Tap, or Swipe!",
              style: TextStyle(color: Colors.white),
            ),
          ),
        ),
```

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

Gesture Example
DEBUG
Tap, Double Tap, or Swipe!

```
Double Tapped!
```

```
Swiped! Offset(2.0, 0.7)
Swiped! Offset(0.7, 0.0)
Swiped! Offset(0.7, 0.0)
Swiped! Offset(0.7, 0.0)
Swiped! Offset(0.7, 0.0)
Swiped! Offset(0.7, 0.0)
Swiped! Offset(0.7, 0.0)
```

Firebase
p2smpl
Project Overview
Project shortcuts
Authentication
App Check
Firestore Database
Genkit NEW
What's new
Vertex AI NEW
Product categories
Build
Run
Analytics
AI
Spark
No-cost ($0/month)
Upgrade NEW
Authentication
Users
Sign-in method
Templates
Usage
Settings
Extensions
The following Authentication features will stop working when Firebase Dynamic Links shuts down on August 25, 2025: email link authentication for mobile apps, as well as Cordova OAuth support for web apps.
Search by email address, phone number, or user UID
Add user
Identifier
Providers
Created
Signed In
User UID
abcd@gmail.com
Mar 25, 2025
Mar 25, 2025
olCFYOlA9CfKrv15iZk4ms5nF...
test4@gmail.com
Mar 25, 2025
Mar 25, 2025
V6X8sh2tKUZtpz6j58hjb6I543...
aaryan@gmail.com
Mar 5, 2025
Mar 19, 2025
D4bRMl0bWJe5ZJ8vW59IDPl...
test@gmail.com
Mar 4, 2025
Mar 25, 2025
E5JPqRZqlLQ9RRiYEMVbcNe...
Rows per page:
50
1 – 4 of 4

Firebase
p2smpl
Project Overview
Project shortcuts
Authentication
App Check
Firestore Database
Genkit NEW
What's new
Vertex AI NEW
Product categories
Build
Run
Analytics
AI
Spark
No-cost ($0/month)
Upgrade NEW
Cloud Firestore
Add database
Ask Gemini how to get started with Firestore
Data
Rules
Indexes
Disaster Recovery NEW
Usage
Extensions
Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing
Configure App Check
Panel view
Query builder
recipes > 1
More in Google Cloud
(default)
Start collection
groceryItems
recipes
recipes
Add document
1
BPBQq0IXsK7Dlw0KaBPi
EZj4x1a33hSeI9vFFq95
GKOr3oazlEstcjZltCPO
ItkL9psz73b7fsewE3rK
OTV9KaMZDOpWITpvZ767
PgqQ7dnvTkyprudGKIx5
1
Start collection
Add field
createdBy: "userId"
imageUrl: "https://media.istockphoto.com/id/12840649
penne-pasta-in-red-sauce.jpg?
s=1024x1024&w=is&k=20&c=fzBBbRyBaCRGy
ingredients
0 "Tomato"

11:39
Login
Email
Password
Login
Don't have an account? Sign Up

11:59
Sign Up
Full Name
Email
Password
Sign Up
1:1

23:22
52%
Recipes
Recommended Recipes
Red Sauce Pasta
Egg Masala
Chicken Steak
Avocado sandwich
Pizza
Pepperoni
Recipes
Grocery List
Settings

23:22
52%
Grocery List
Dairy
Butter
Meat
Chicken-Breast
Veggies
Bell-Peppers
Carrots
Tomatoes
Recipes
Grocery List
Settings

Install app
PWA Tutorial
127.0.0.1:5500
Install
Cancel

PWA Tutorial - Space Exploration Info
127.0.0.1:5500
Minimize
Space Exploration
Discover the fascinating world of space exploration—from historic moon landings to future missions to Mars and beyond.

30°C
Smoke
Search
ENG
IN
09:32
02/04/2025

PWA Tutorial

PWA
images
index.html
manifest.json
offline.html
serviceworker.js

127.0.0.1:5500/index.html
Open in app
Space Exploration
Discover the fascinating world of space exploration—from historic moon landings to future missions to Mars and beyond.
Elements
Console
Sources
Network
Performance
Preserve log
Disable cache
Fast 4G
Filter
Invert
More filters
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Big request rows
Group by frame
Overview
Screenshots
20 ms
40 ms
60 ms
80 ms
100 ms
Recording network activity...
Perform a request or hit Ctrl + R to record the reload.
Learn more
Console
What's new
AI assistance
Search
top
Filter
Default levels
No Issues
http://127.0.0.1:5500/manifest.json serviceworker.js:12
http://127.0.0.1:5500/images/app-icon.png serviceworker.js:12
Service Worker registered successfully: http://127.0.0.1:5500/ index.html:35
Service Worker registered successfully. index.html:47
http://127.0.0.1:5500/index.html serviceworker.js:12
http://127.0.0.1:5500/manifest.json serviceworker.js:12
30°C
Smoke
Search
ENG
IN
09:52
02/04/2025

MAD&PWA-24-25 D15B
Exp_8 Guidelines with ex
PWA Setup Guide
mpl8 - Google Docs
Space Exploration Info
127.0.0.1:5500/index.html
Open in app
Space Exploration
Discover the fascinating world of space exploration—from historic moon landings to future missions to Mars and beyond.
Elements
Console
Sources
Network
Performance
Memory
Application
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5500/
Network requests
Update
Unregister
Source serviceworker.js
Received 4/2/2025, 9:52:33 AM
Status #494 activated and is running
Stop
Clients http://127.0.0.1:5500/index.html
Push Test push message from DevTools.
Push
Sync test-tag-from-devtools
Sync
Periodic sync test-tag-from-devtools
Periodic sync
Update Cycle
Version
Update Activity
Timeline
#494
Install
Manifest
Service workers
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
pwa-cache-v1 - htt...
pwa - http://127.0....
Storage buckets
Console
What's new
AI assistance
Search
top
Filter
Default levels
No Issues
http://127.0.0.1:5500/manifest.json serviceworker.js:12
http://127.0.0.1:5500/images/app-icon.png serviceworker.js:12
Service Worker registered successfully: http://127.0.0.1:5500/ index.html:35
Service Worker registered successfully. index.html:47
http://127.0.0.1:5500/index.html serviceworker.js:12
http://127.0.0.1:5500/manifest.json serviceworker.js:12
30°C
Smoke
Search
ENG
IN
09:57
02/04/2025

MAD&PWA-24-25 D15B
Exp_8 Guidelines with ex
PWA Setup Guide
mpl8 - Google Docs
Space Exploration Info
127.0.0.1:5500/index.html
Open in app
Space Exploration
Discover the fascinating world of space exploration—from historic moon landings to future missions to Mars and beyond.
Elements
Console
Sources
Network
Performance
Memory
Application
Filter by path
http://127.0.0.1:5500
Origin http://127.0.0.1:5500
Bucket name default
Is persistent No
Durability relaxed
Quota 0 B
Expiration None
Name Respo... Conte... Conte... Time C... Vary H...
0 / basic text/ht... 3,244 4/2/20... Origin
1 /index.html basic text/ht... 3,244 4/2/20... Origin
2 /offline.html basic text/ht... 1,726 4/2/20... Origin
Total entries: 3
Manifest
Service workers
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
pwa-cache-v1 - htt...
pwa - http://127.0....
Storage buckets
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking miti...
Notifications
Payment handler
Periodic background ...
Console
What's new
AI assistance
Search
top
Filter
Default levels
No Issues
30°C
Smoke
Search
ENG
IN
09:57
02/04/2025

Google Docs
Space Exploration Info
Open in app
Sources
Network
Performance
Memory
Application
Service workers
Offline
Update on reload
Bypass for network
http://127.0.0.1:5500/
Network requests
Update
Unregister
Source
serviceworker.js
Received 4/2/2025, 9:52:33 AM
Status
#494 activated and is running
Stop
Clients
http://127.0.0.1:5500/index.html
Push
Test push message from DevTools.
Push
Sync
test-tag-from-devtools
Sync
Periodic sync
test-tag-from-devtools
Periodic sync
Update Cycle
Version
Update Activity
Timeline
#494
Install
AI assistance
Search
Filter
Default levels
No Issues

MAD&PWA-24-25 D15B
Exp_8 Guidelines with ex
PWA Setup Guide
mpl8 - Google Docs
PWA Tutorial
127.0.0.1:5500/index.html
You're offline
Elements
Console
Sources
Network
Performance
Preserve log
Disable cache
Offline
Filter
Invert
More filters
All
Fetch/XHR
Doc
CSS
JS
Font
Img
Media
Manifest
WS
Wasm
Other
Big request rows
Group by frame
Overview
Screenshots
20 ms
40 ms
60 ms
80 ms
100 ms
Recording network activity...
Perform a request or hit Ctrl + R to record the reload.
Learn more
30°C
Smoke
Search
ENG
IN
09:51
02/04/2025

Name:Aaryan Kalbhor
Div:D15B
Roll No:26

MAD lab 10

Steps to Deploy an E-commerce PWA to GitHub Pages

1. **Create a GitHub Repository** – Set up a new public repository on GitHub.
2. **Initialize Git** – Link your local project to the repository and commit your code.
3. **Push Code to GitHub** – Upload your project files to GitHub.
4. **Enable GitHub Pages** – Go to the repository's **Settings > Pages**, set the source to the `main` branch, and save.
5. **Install Deployment Tool** – Add the necessary package for GitHub Pages deployment.
6. **Update Project Settings** – Modify the project configuration to specify the deployment URL.
7. **Deploy the PWA** – Build the project and publish it to GitHub Pages.
8. **Access Live Site** – Visit the provided GitHub Pages link to view your deployed PWA.

YouTube
Pages
github.com/Aaryank1337/pwa/settings/pages
Aaryank1337 / pwa
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Webhooks
Environments
Codespaces
Pages
Security
Code Security
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
GitHub Pages is currently disabled. Select a source below to enable GitHub Pages for this repository. Learn more about configuring the publishing source for your site.
None
Save
Visibility
GitHub Enterprise
With a GitHub Enterprise account, you can restrict access to your GitHub Pages site by publishing it privately. You can use privately published sites to share your internal documentation or knowledge base with members of your enterprise. You can try GitHub Enterprise risk-free for 30 days. Learn more about the visibility of your GitHub Pages site.
Start free for 30 days
Search
ENG
IN
21:37
08/04/2025

GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
Your site is live at https://aaryank1337.github.io/pwa/
Last deployed by Aaryank1337 1 minute ago
Visit site
Build and deployment
Source

YouTube
Space Exploration Info
aaryank1337.github.io/pwa/
🚀 Space Exploration
Explore missions, history, and the future of space travel
Historic Moon Landing
The Apollo 11 mission landed the first humans on the Moon in 1969, marking a significant achievement in space exploration.
Future Missions to Mars
NASA and other agencies are planning manned missions to Mars, aiming to establish human presence on the Red Planet by the 2030s.
Search
ENG
IN
21:42
08/04/2025

Name : Aaryan Kalbhor
Div : D15B
Roll No : 26

MAD 11

Steps to Use Google Lighthouse for PWA Analysis

1. **Open Chrome DevTools** – Open Google Chrome and press F12 or right-click on the page and select **Inspect**.
2. **Go to Lighthouse Tab** – Navigate to the **Lighthouse** tab in DevTools.
3. **Select Audit Criteria** – Choose **Performance, PWA, Accessibility, Best Practices**, and other relevant metrics.
4. **Run the Audit** – Click the **Generate Report** button and wait for the analysis to complete.
5. **Review Results** – Check the generated report, which includes PWA compliance, performance, and accessibility insights.
6. **Make Necessary Changes** – Modify **theme color, icons, manifest.json**, and other settings to improve PWA compliance.
7. **Re-run Lighthouse** – Test again to verify improvements and ensure all PWA requirements are met.

Elements
Console
Sources
Network
Performance
Lighthouse
1
(new report)
Generate a Lighthouse report
Analyze page load
Mode
Learn more
Navigation (Default)
Timespan
Snapshot
Device
Mobile
Desktop
Categories
Performance
Accessibility
Best practices
SEO

Elements
Console
Sources
Network
Performance
Memory
Lighthouse
9:49:12 PM - aaryank1337.git
https://aaryank1337.github.io/pwa/
100
Performance
100
Accessibility
96
Best Practices
90
SEO
100
Performance
Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.
0–49
50–89
90–100
Space Exploration
Explore missions, history, and the future of space travel
Historic Moon Landing
The Apollo 11 mission landed the first humans on the Moon in 1969, marking a significant achievement in space exploration.
Future Missions to Mars
NASA and other agencies are planning manned missions to Mars, aiming to establish human presence on the Red Planet by the 2030s.

METRICS

Expand view

- First Contentful Paint
  0.5 s
- Largest Contentful Paint
  0.5 s
- Total Blocking Time
  0 ms
- Cumulative Layout Shift
  0.005
- Speed Index
  0.5 s

View Treemap

DIAGNOSTICS

- Eliminate render-blocking resources — Potential savings of 100 ms
- Minify JavaScript — Potential savings of 154 KiB
- Avoid serving legacy JavaScript to modern browsers — Potential savings of 9 KiB
- Reduce unused JavaScript — Potential savings of 713 KiB
- Avoid large layout shifts — 1 layout shift found
- User Timing marks and measures — 4 user timings
- Initial server response time was short — Root document took 20 ms
- Avoids enormous network payloads — Total size was 48 KiB
- Avoids an excessive DOM size — 11 elements

Aaryan Kalthar
D15B - 26

A+ 04

MAD Assignment - 1

Q1)

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

Ans. 1. Single codebase for multiple platforms. Write one codebase for both Android and IOS, reducing development effort and maintanence.

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 perfomance apps

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

Advantages

① Faster development Time : Hot reload and single codebase reduces development time significantly.

② Cost effective : Since the same code runs on both Android and IOS, business save on development and maintainence cost.

③ Reduced performance issues : The app runs natively without relying on intermediate bridges like in react native reducing lag.

b. Discuss how the flutter framework differ from traditional approaches and why it has gained popularity in the developer community.

Ans 1) Single codebase vs separate codebase

- Traditional Approach : Developers need to write separated 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 vs Native UI Components.

→ Traditional Approach : Relies on platform native UI component which can lead to inconsistencies and performance issues

Flutter : Uses the Skia rendering engine to draw everything from scratch ensuring a consistent UI acrosses 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 : Business 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 access to GPO rendering ensure smooth animations and high performance

(5) Easy Integration with Backend Technologies : works well with Firebase, REST API's GraphQL, and other backend technologies simplifying full stack development.

Q2)

a. Describe the concept of widget tree in flutter. Explain how widget composition is used to build wear interfaces.

Ans widget tree in flutter.

In flutter, the widget tree is the fundamental structure that represent the UI of an application. It is a hierachial arrangement of widgets where each widget define a part of user interface. Flutter's UI is entirely built using widgets which can be stateless or stateful. The widget tree determines how the UI is rendered and updated when changes occur.

— widget Composition in flutter

Widget composition refers to building complex UIs by combining smaller, reusable widgets. Instead of creating large, monolithic UI components, flutter encourages breaking the UI into smaller manageable widgets that can be reused and nested with each other.

Eg:- 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. Maintability : Breaking UI into smaller widgets makes it easier to debug and update.
3. Performance :- Flutter efficiently rebuilds only the necessary part of widget tree.

Sundaram
FOR EDUCATIONAL USE

b. Provide example of commonly used widgets and their role in creating a widget tree.

→ ① Structural widgets

These widgets act as the foundation 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 for body floating action button etc
- Container :- A versatile widget used for styling, padding, margin & background customization

Eg:- MaterialApp(
home : Scaffold (
appBar : AppBar (title : Text ('Flutter widget Tree')),
body : Container (
padding : EdgeInsets.all (16.0),
child : Text ("Hello"),
),
),
),

② Input & Interaction widgets

Textfield - Accepts text input from users
Elevation Button - A button with elevation
Gesture Detector - Detects gesture like taps, swipes and long presses.

eg:- Column (
children : [
TextField (decoration : InputDecoration (labelText : "N
ElevatedButton (
onPressed : () {
print ("Button pressed");
},
child : Text ("submit");
),
],
);

(3) Display & Styling widget

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

eg: Column (
children : [
Text ("Welcome", style : TextStyle (fontSize : 24, fontWeight : FontWeight.bold)),
Image.network ("https://flutter.dev/images/flutter-logo-sharing.png"),
],
);

Q3)

a) Discuss the importance of state management in flutter applications.

→ In flutter, state refers to data that can change during the lifetime of application. This include:

- 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) Appwide Status - Data shared across multiple widgets. ~~Impor~~

* Importance of state management:

- Efficient UI update: Flutter's UI is rebuilt whenever state changes. Efficient state mangement ensure that only necessary widgets are updated.
- Code Maintainability & Scalability: Managing state properly makes the code modular, redable and scalable for larger application
- Data consistency and synchronization

Proper state management ensures that data remain ~~co~~ constant across diff screens & widgets.

FOR EDUCATIONAL USE

Sundaram

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

→ <u>Set State</u> - Local state

Pros - simple built in easy to use

Cons - Not scalable causes unnecessary re-renders

Best-Use Cases - Small UI updates (eg, counter, toggle switch)

<u>Provider</u> - Appwide state

Pros - lightweight, recommended by flutter, efficient

Cons - Boilerplate code for nested providers

Best-Use Cases - Medium Scal App (eg. Authentication, theme, API data)

Riverpod :- Appwide State (More Scalable than provider)

Pros :- Eliminates provider's limitations, improves performance

Cons - requires learning new concepts

Best usecases - Large apps needing global state (eg shopping cart, user sessions)

Scenarios for Each Approach

1. Use setState when managing simple UI elements with in a single widget
2. Use Provider when sharing state across multiple widgets
3. Use Riverpod when building a complex, scalable app with global state management

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

→ Firebase provides a powerful backend solution for flutter application offering services like authentication, real time database, cloud functions, storage and more.

* Steps to Integrate.

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.

② Register the flutter app with firebase
- In firebase project dashboard click "Add App" and select Android or ios based on your platform
- for Android : Enter Android package name and download the google service .json file and place it in android/app/
- for ios : Enter the ios bundle identifier Download the GoogleService-Info.plist-file and place it in ios/runner

③ Install firebase dependencies
Add firebase dependencies in pubspec.yml.
firebase core
firebase auth
cloud_firestore. Run: flutter pub get.

FOR EDUCATIONAL USE

(4) Configure Firebase for Android & iOS

for Android

1) Open Android/build.gradel and ensure the following
classpath 'com.google.gms : google-services : 4.3.10'

2) Open android/app/build.gradle and add the
bottom apply plugin : 'com.google.gms.google-services'

(5) Initialise Firebase in flutter

```
void main() async {
  widgetFlutterBinding.ensureInitialised();
  await Firebase.initializeApp();
  runApp (MyApp());
```

Benefits of using firebase

Firebase is a Backend as - a service (BaaS) that simplifies backend development for flutter Apps. Here are some key benefits.

1.) Easy to set up & Scale
No need to manage backend infrastructure
Scale automatically based on usage.

2) Authentication
Provides email/password, Google, facebook and phone authentication
Seamless integration with Firebase Authentication

3) Cloud Storage
Secure file-storage for img, videos & documents

4) Push Notification (Firebase ~~could~~ cloud Messaging)
send real-time notification to users across different platforms

Sundaram
FOR EDUCATIONAL USE

b) Highlight the firebase service 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 and Apple.

2) Cloud Firebase

Stores and syncs data in real time across devices support structured data, queries and offline access.

Eg :- Firebase Firestore.instance.collection('users').add({
'name' : 'Aaryan',
'email' : 'test@gmail.com',
});

3) Realtime Database

A realtime, json based database that automatically updates data across devices

ex. DatabaseReference ref = FirebaseDatabase.instance.ref("message");
ref.set({"text" : "Hello."});

4) Firebase Cloud Messaging (FCM)

Enables push notifications and messaging between users

Ex. FirebaseMessaging.instance.subscribeToTopic("news");

5) Firebase Analytics

Tracks user interaction and app performance

Ex. Firebase Analytics analytics. Firebase Analytics. Instance analytics. log Event (name: "button - clicked", parameters: { "button": "subscribe" });

6) Firebase Hosting

Deploys and serves web application securely with automate SSL

* Data Synchronization in Firebase

Firebase ensures real-time data synchronization across multiple devices and platforms using firestore and realtime Database.

1. Cloud Firestore Sync Mechanism

Uses realtime listeners to update UI instantly when data changes

Eg:-
```
firebase firestore .instances .collection ('users') .snapshot()
                                                listen ((snapshot) {
    for ( var doc in snapshot.docs){
        print (doc ['name']);
        }
    });
```

2 Realtime Database Sync Mechanism
Uses persistent WebSocket connection for live update.
Ex. DatabaseReference ref = FirebaseDatabase.instance.ref("messages");

```
ref.onValue.listen((event){
    print(event.snapshot.value);
});
```

3 Offline Data Sync
Firestore cache data locally and syncs changes when the device is online
Eg: FirebaseFirestore.instance.settings(persistenceEnabled: true);

4 Cloud Functions for automated updates
Automates backend logic to trigger updates when data changes.

FOR EDUCATIONAL USE

Aaryan Kalbhor
DISB-26

05/05

MPL Assignment - 2

Define progressive webApp (PWA) and explain its significance in modern web development. Discuss the key characteristics that differentiate PWA's from traditional mobile apps

A progressive webApp (PWA) is a type of web application that works like a mobile app but runs in browser. It can be installed on a device works offline and provides a fast and a smooth user experience

Significance of PWA in modern web development

- Cross platform compatibility
- offline support
- fast performance
- No app store required
- lower development cost

Key difference between PWA & traditional mobile App

Features	PWA	Traditional Mobile Apps
Installation	Direct from browser	Download from App Store
Internet Required	Works offline with caching	Usually requires internet
Performance	Fast with service workers	Faster but needs Installation
Updates	Automatic no appstore approval	manual update needs
Development cost	lower	Higher

FOR EDUCATIONAL USE

Q2) Define responsiveness web design and explain its importance in the context of progressive web Apps compare and contrast responsiveness fluid & adaptive web design approaches

→ Responsive web design (RWD) is technique that makes web pages adjust automatically to different screen sizes and devices. It ensures a good user experience on mobiles, tablets and desktops without needing separate version of website.

* Importance of responsiveness design in PWA.

- Better user experience - PWA's work smoothly on any device.
- Faster load time - optimised design improves speed
- SEO benefits - Google ranks responsive sites higher
- Cost effective - No need to build multiple versions for different screens.

* Comparison of web design approaches

Approaches	How it works	Pros	Cons.
- Responsiveness	Uses flexible grids and CSS media queries to adjust layout	works on cell. devices	Can be complex to design
- Fluid	Uses parent based widths instead of fixed pixels so. elements resize smoothly	works well on diff screen sizes, easy to implement	less control over layout on large screens
- Adaptive	Uses fixed layouts that change at specific breakpoints	optimized for known screen sizes	more efficient required to design for cab screen size.

Sundaram
FOR EDUCATIONAL USE

Key differences

- Responsive adapts dynamically to all screens.
- Fluid resizes smoothly, but may be fully a prioritised
- Adaptive load different layouts based on device types.

Describe the lifecycle of service worker including to registration installation and activation phase

Lifecycle of service worker.

A service worker is a script that runs in background and helps a web app work offline load faster and send push notification. Its lifecycle has three main phases.

1) Registration Phase

The browser registers the source worker using Javascript.

Code Eg:-

```
If ('service worker' in navigator) {
    navigator service worker. register ('/sw.js')
  . then (() => console.log ('service worker 'registered'))
  . catch (error) => console.log ('registration failed', error)
```

This tells the browser to install and activate the service worker.

2) Installation phase

The source worker downloads necessary files (HTML, CSS, JS) and stores them in cache.

If successful it moves to activation phase.

Eg:

```
Self add EventListener ('install event → {
        event. wait until (
```

FOR EDUCATIONAL USE

```
. caches.open (app.cache) .then (cache → {
    return cache.addAll ('[ /', 'index.html', 'styles.cs
    });
});
});
```

This ensure the app load even without internet.

3) Activation phase

- The old service worker is replaced with new one.
- Unused cache files from the previous version are deleted

Final step fetch & sync

One activated the service worker intercepts network requests serves cached files and sync data when the Internet is available. This lifecycle make PWA's faster, more reliable and capable of working offline.

Q4) Explain the use of Indexed DB in the system service worker of data storage

Ans Use of Indexed DB. in service worker for data storage

Indexed DB is a browser database that stores large amounts of structured data like json objects. It helps PWA's work offline by saving and retrieving data efficiently.

Why use Indexed DB. in service worker?

1) Office Support - Stores data when offline and sync is later
2) Efficient storage - Saves structured data like user setting cost items or forms inputs.
3) Faster Access - Retrieves data quickly without needing a network request.

Persistent Data - Data remains saved even after the browser is closed.

How service workers uses Indexed DB?

Opening the database

```
let db.
let request = IndexDB.open ('My database');
  request.onsuccess = function (event) {
      db = event.target.result;
  };
```

Creating a store & Adding Data.

```
request.onupgradeneeded = function (event) {
  let db = event.target.result;
  let store = db.createObjectStore ('users' & keypath : 'id')
store.add ({id : 1 , name : 'John Doe' , age : 25 });
  };
```

Fetching Data in service worker.

```
let transaction = db.transaction ('users' : 'readonly');
let store = transaction.objectStore ('users');
let getuser = store.get (1).

getusers.onsuccess = function () {
  console.log ('get user detail');
  };
```