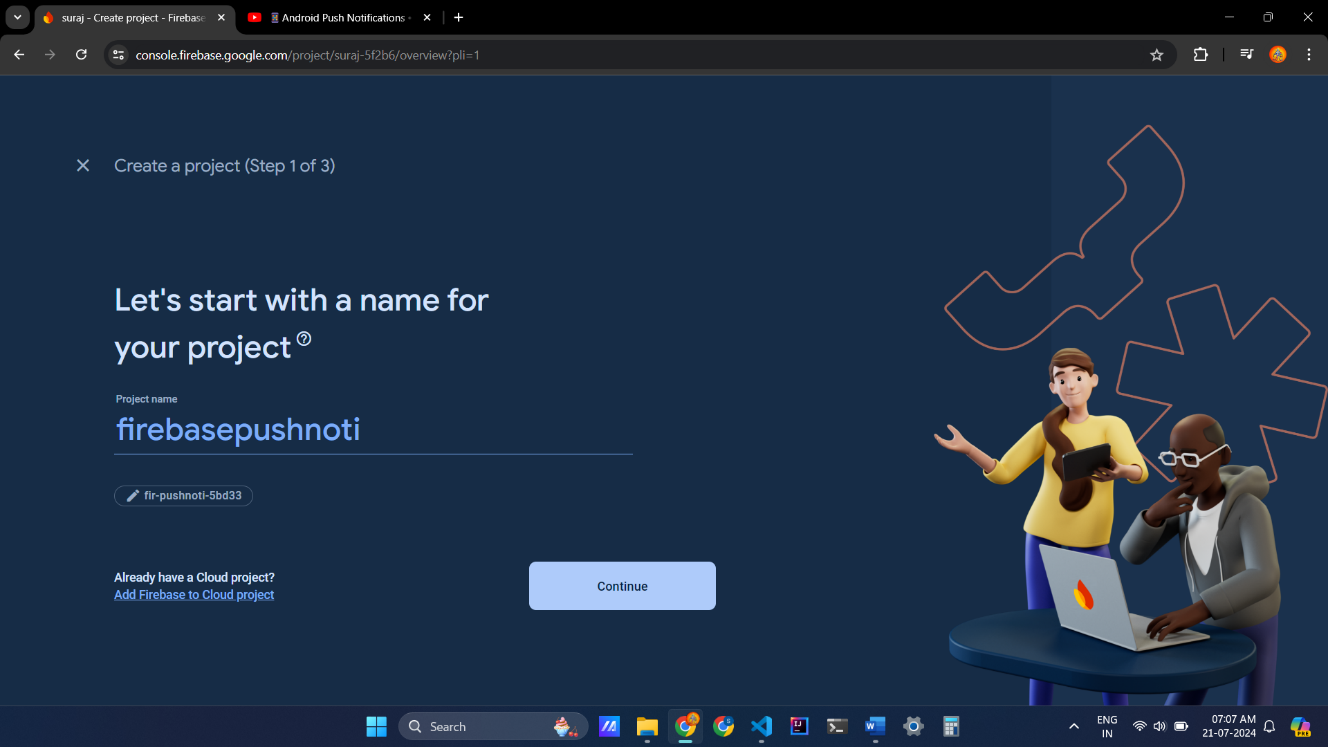
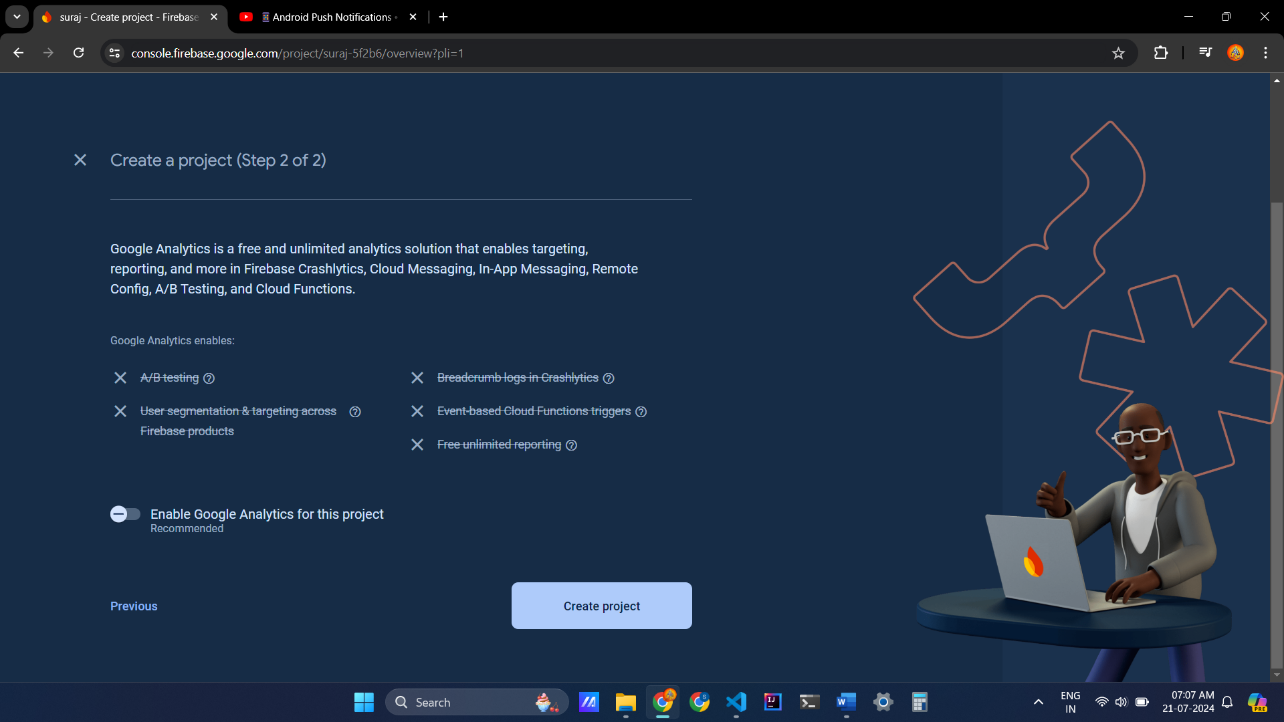
Firebase push notification

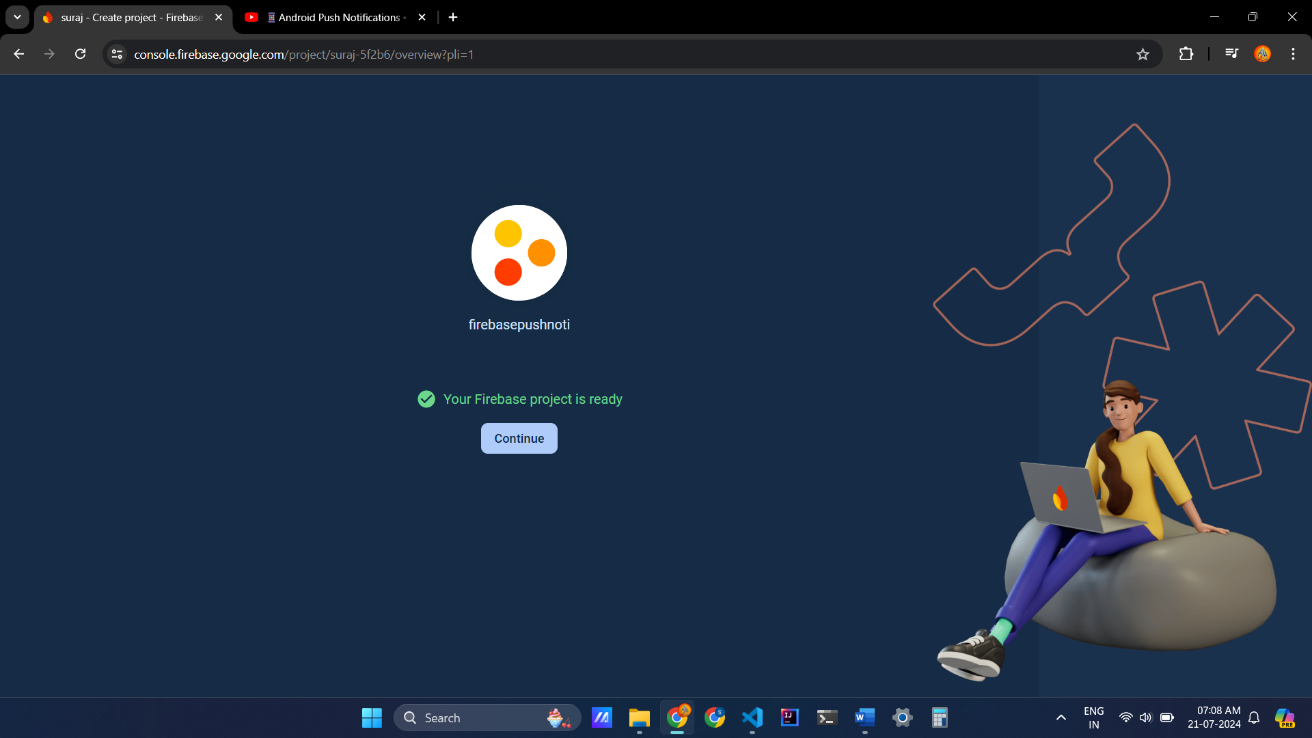
1. Create brand new flutter project
2. Now creating firebase project
3. Create project with name



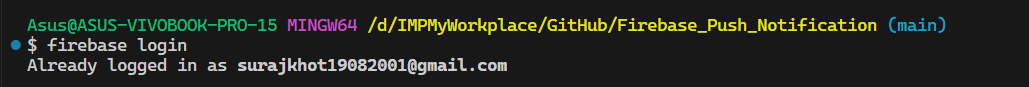
1. Disable google analytics



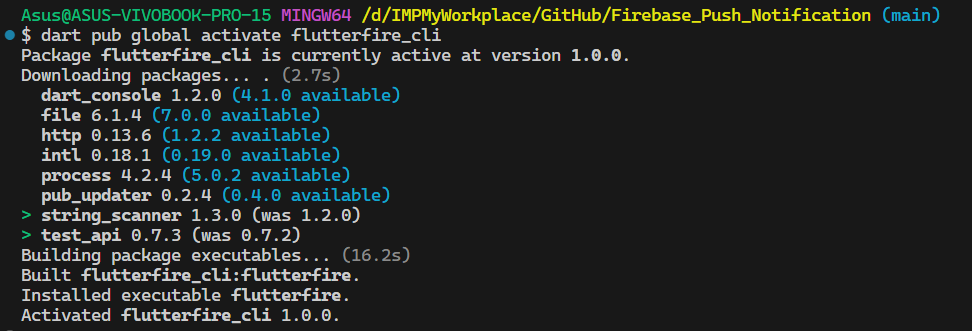
1. Done with project



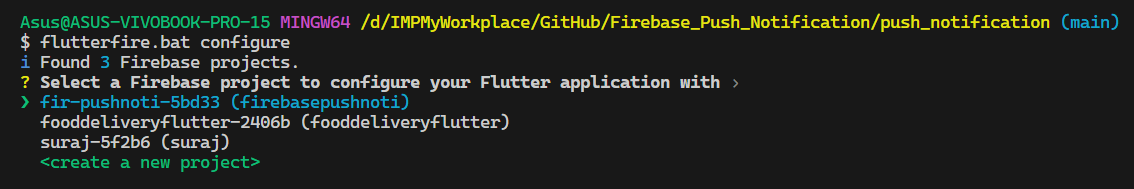
1. Now edit flutter code to connect with firebase project using cli
2. firebase login first



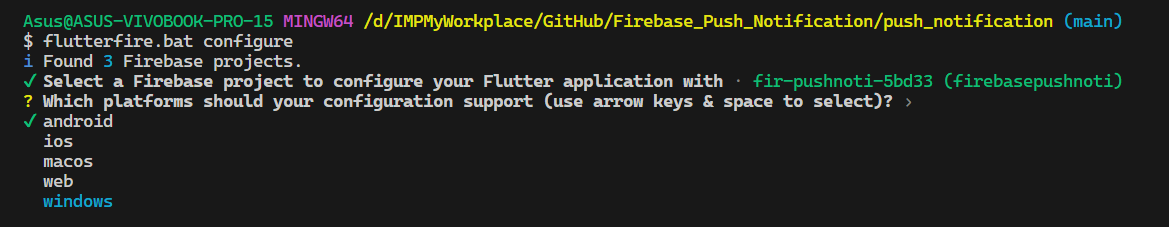
1. dart pub global activate flutterfire\_cli to activating firebase cli

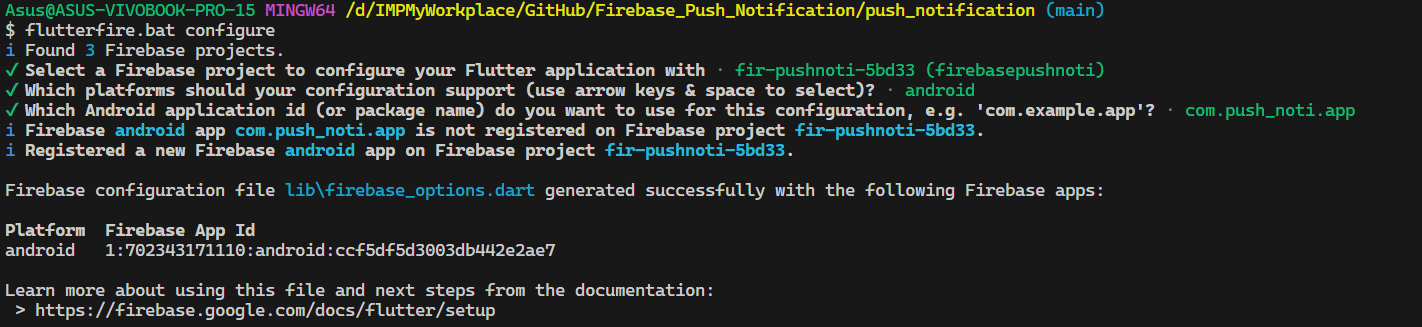


1. flutterfire.bat configure to select the project and add to flutter



Change directory to project directory





1. now add dependencies
2. flutter pub add firebase\_core
3. flutter pub add firebase\_messaging
4. now to add project then we use main.dart

import 'package:firebase\_core/firebase\_core.dart';

import 'package:flutter/material.dart';

import 'package:push\_notification/firebase\_options.dart';

import 'package:push\_notification/home.dart';

void main() async {

  WidgetsFlutterBinding.ensureInitialized();

  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);

  runApp(const MyApp());

}

class MyApp extends StatelessWidget {

  const MyApp({super.key});

  @override

  Widget build(BuildContext context) {

    return const MaterialApp(

      home: Home(),

    );

  }

}

WidgetsFlutterBinding:

A concrete binding for applications based on the Widgets framework.

This is the glue that binds the framework to the Flutter engine.

When using the widgets framework, this binding, or one that implements the same interfaces, must be used. The following mixins are used to implement this binding:

* [GestureBinding], which implements the basics of hit testing.
* [SchedulerBinding], which introduces the concepts of frames.
* [ServicesBinding], which provides access to the plugin subsystem.
* [PaintingBinding], which enables decoding images.
* [SemanticsBinding], which supports accessibility.
* [RendererBinding], which handles the render tree.
* [WidgetsBinding], which handles the widget tree.

initializeApp:

Initializes a new [FirebaseApp] instance by [name] and [options] and returns the created app. This method should be called before any usage of FlutterFire plugins.

The default app instance can be initialized here simply by passing no "name" as an argument in both Dart & manual initialization flows. If you have a google-services.json file in your android project or a GoogleService-Info.plist file in your iOS+ project, it will automatically create a default (named "[DEFAULT]") app instance on the native platform. However, you will still need to call this method before using any FlutterFire plugins.

DefaultFirebaseOptions:

Default [FirebaseOptions] for use with your Firebase apps.

Example:

import 'firebase\_options.dart';  
// ...  
await Firebase.initializeApp(  
options: DefaultFirebaseOptions.currentPlatform,  
);

1. We just forgot to add package name correctly while initializing firebase project

For that just go to

google-services.json and change package name with error in package name

  "client": [

    {

      "client\_info": {

        "mobilesdk\_app\_id": "1:702343171110:android:ccf5df5d3003db442e2ae7",

        "android\_client\_info": {

//here made change

          "package\_name": "com.example.push\_notification"

        }

      },

      "oauth\_client": [],

      "api\_key": [

        {

          "current\_key": "AIzaSyAXPez3JH\_YwYVOhJV82PtF46FGKn\_gFUQ"

        }

      ],

1. What is our plan to create
2. Create api named folder in lib
3. Now create firebase\_api.dart named class file
4. Make a plan

class FirebaseApi{

  //create instance of firebase messaging

  //function to initialize notification

  //function to handle received notification

  //function to initialize foreground and background settings

}

1. Go with plan

import 'package:firebase\_messaging/firebase\_messaging.dart';

import 'package:flutter/material.dart';

class FirebaseApi {

  //create instance of firebase messaging

  final firebaseMessaging = FirebaseMessaging.instance;

  //function to initalize notification

  Future<void> initNotification() async {

    //prompt user to allow notification

    await firebaseMessaging.requestPermission();

    //fetch the FCM token (firebase cloud messaging)

    final fCMToken = await firebaseMessaging.getToken();

    //print token

    debugPrint("token:  $fCMToken");

  }

  //function to handel recevied notification

  //function to initalize forground and background settings

}

1. Now to test call this function in main.dart

import 'package:firebase\_core/firebase\_core.dart';

import 'package:flutter/material.dart';

import 'package:push\_notification/api/firebase\_api.dart';

import 'package:push\_notification/firebase\_options.dart';

import 'package:push\_notification/home.dart';

void main() async {

  WidgetsFlutterBinding.ensureInitialized();

  //initalizing app

  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);

  //calling our api's notification method here

  await FirebaseApi().initNotification();

  runApp(const MyApp());

}

class MyApp extends StatelessWidget {

  const MyApp({super.key});

  @override

  Widget build(BuildContext context) {

    return const MaterialApp(

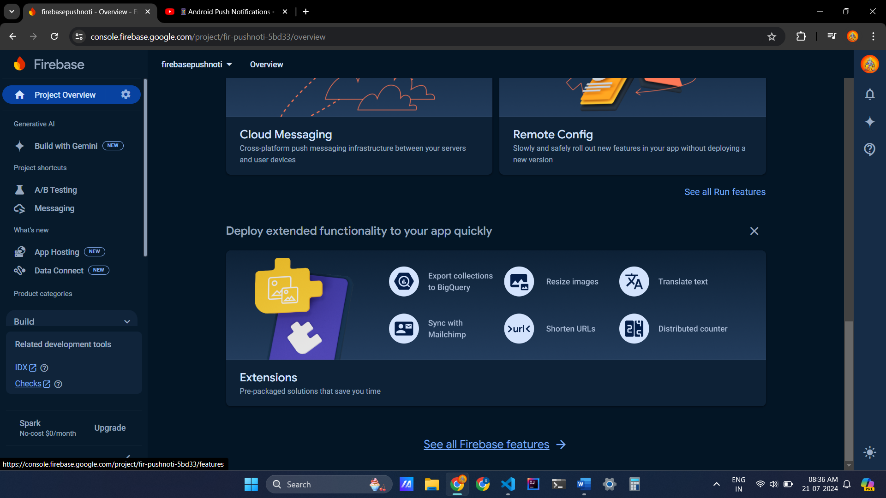
      home: Home(),

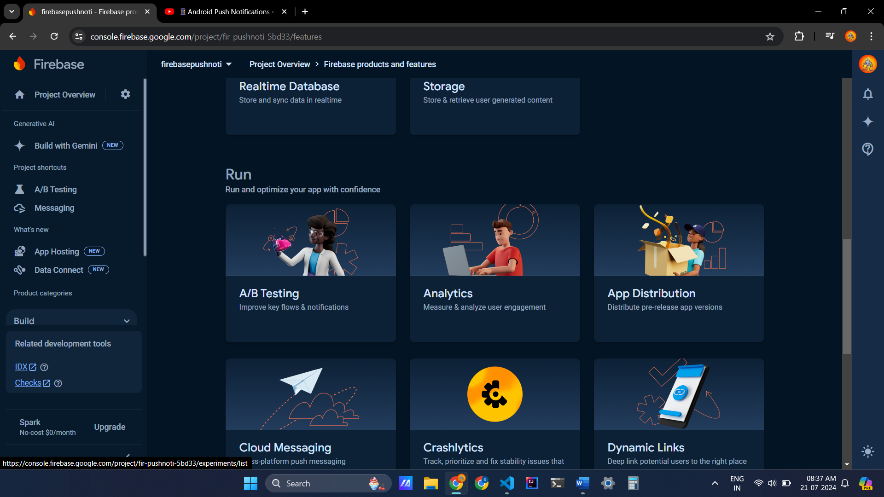
    );

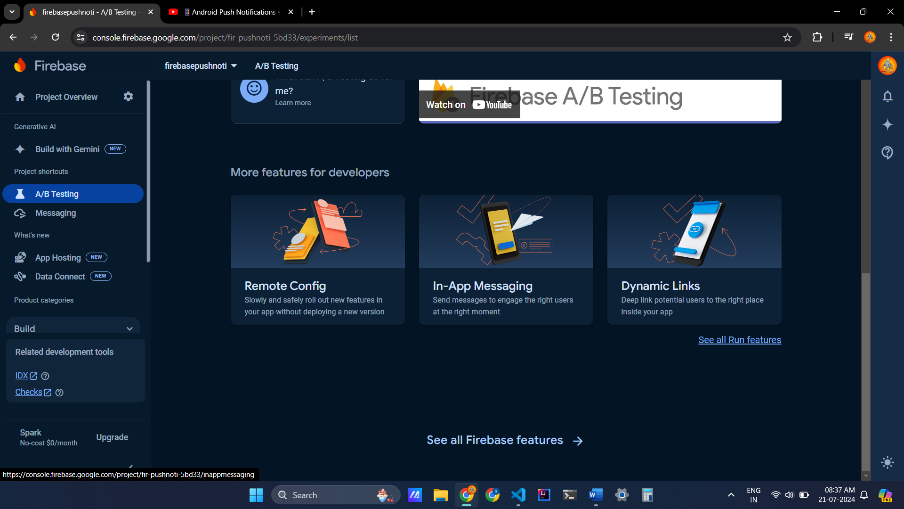
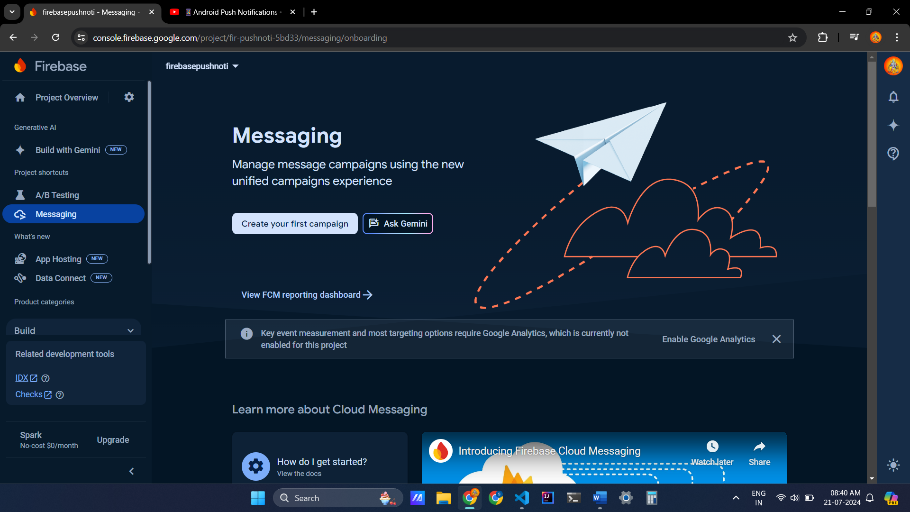
  }

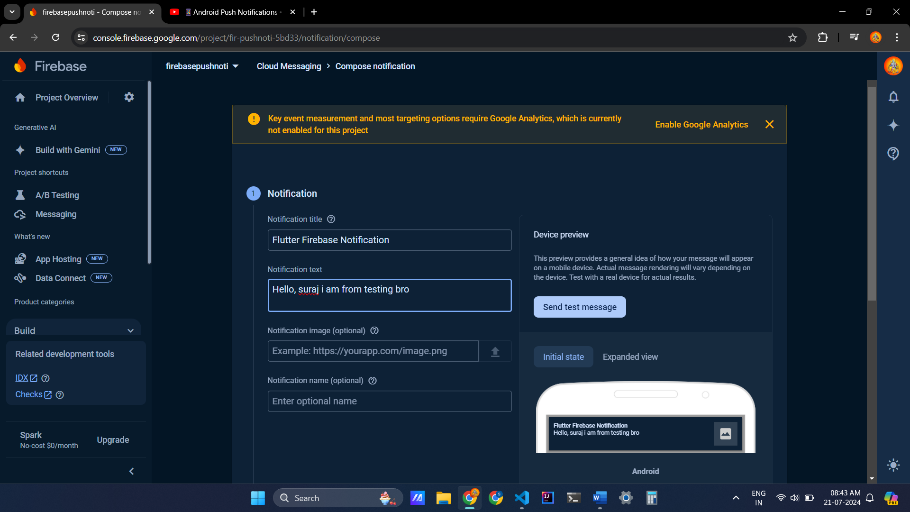
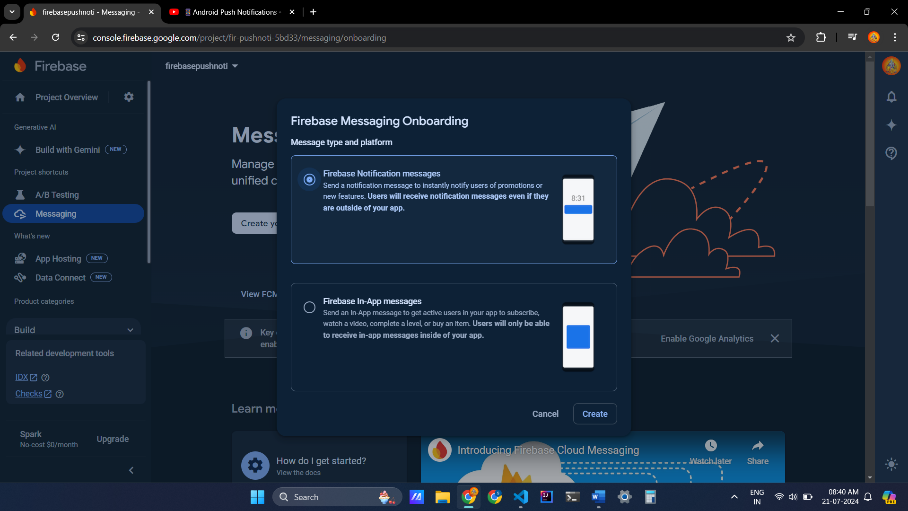
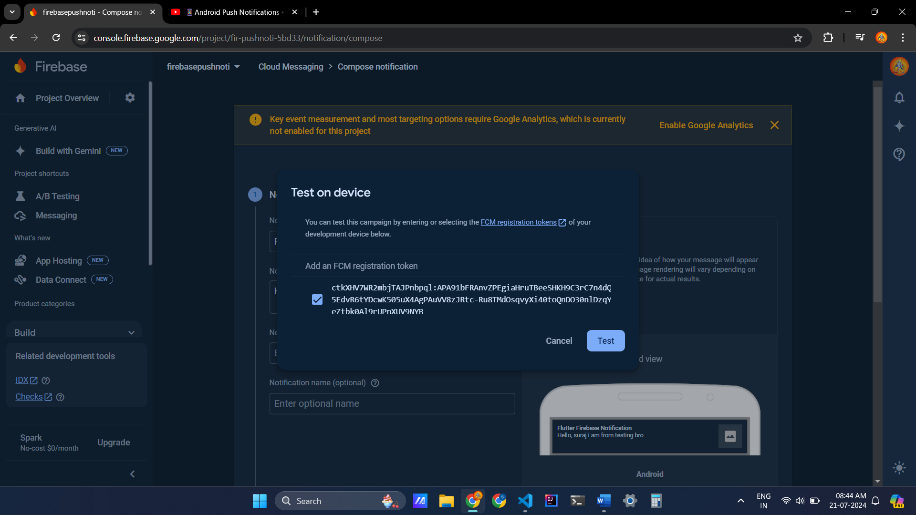
}

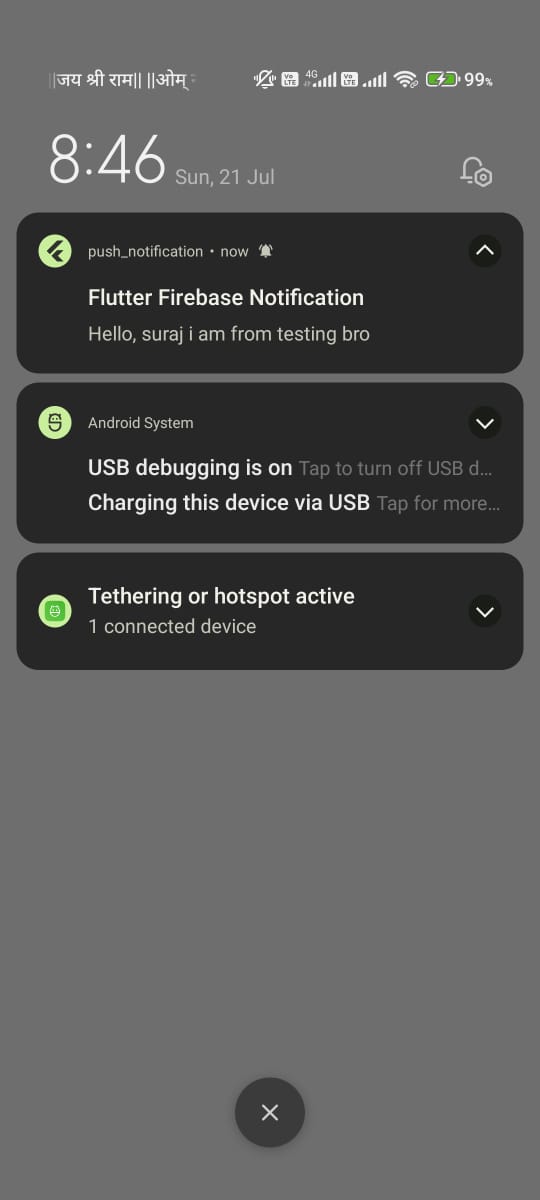
1. Now to test go with token
2. Go to firebase project
3. Go to all firebase features
4. Go to A/B testing
5. Now scroll down for in-app messaging feature
6. Now click on create your first campaign.
7. Select notification message
8. No add fields
9. Now click on send message
10. Now add token
11. And then click on test









1. Now navigating to custom page when tap on notification to do like this
2. Define route key and pass routes as there in main.dart

import 'package:firebase\_core/firebase\_core.dart';

import 'package:flutter/material.dart';

import 'package:push\_notification/api/firebase\_api.dart';

import 'package:push\_notification/firebase\_options.dart';

import 'package:push\_notification/home.dart';

import 'package:push\_notification/notification\_screen.dart';

//The state for a [Navigator] widget.

final naviKey = GlobalKey<NavigatorState>();

void main() async {

  WidgetsFlutterBinding.ensureInitialized();

  //initalizing app

  await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);

  //calling our api's notification method here

  await FirebaseApi().initNotification();

  runApp(const MyApp());

}

class MyApp extends StatelessWidget {

  const MyApp({super.key});

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      home: const Home(),

      //key

      navigatorKey: naviKey,

      //routes

      routes: {

        '/notification\_screen': (context) => const NotificationScreen(),

      },

    );

  }

}

1. Now edit firebase\_api.dart

import 'package:firebase\_messaging/firebase\_messaging.dart';

import 'package:flutter/material.dart';

import 'package:push\_notification/main.dart';

class FirebaseApi {

  //create instance of firebase messaging

  final firebaseMessaging = FirebaseMessaging.instance;

  //function to initalize notification

  Future<void> initNotification() async {

    //prompt user to allow notification

    await firebaseMessaging.requestPermission();

    //fetch the FCM token (firebase cloud messaging)

    final fCMToken = await firebaseMessaging.getToken();

    //print token

    debugPrint("token:  $fCMToken");

    //initalize for further push notification

     initPushNotification();

  }

  //function to handel recevied notification

  void handleNoti(RemoteMessage? message) {

    //if message is null

    if (message == null) return;

    //if having data navigate to noti screen when click on notification

    naviKey.currentState!.pushNamed('/notification\_screen', arguments: message);

  }

  //function to initalize forground and background settings

  Future initPushNotification() async {

    //handle the notification if the app is terminated and now opened

    await FirebaseMessaging.instance.getInitialMessage().then(handleNoti);

    //add event listener when notification opens app

    FirebaseMessaging.onMessageOpenedApp.listen(handleNoti);

  }

}

1. And now create noti screen

import 'package:firebase\_messaging/firebase\_messaging.dart';

import 'package:flutter/material.dart';

class NotificationScreen extends StatelessWidget {

  const NotificationScreen({super.key});

  @override

  Widget build(BuildContext context) {

    final noti = ModalRoute.of(context)!.settings.arguments as RemoteMessage;

    return Scaffold(

      appBar: AppBar(),

      body: Center(

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          children: [

            Text(noti.notification!.title.toString()),

            Text(noti.notification!.body.toString()),

            Text(noti.data.toString()),

          ],

        ),

      ),

    );

  }

}

1. Now test

Congratulations success >>>>>>>>>>>>>>>>>>>>>>>>>.

1. Now to deploy this
2. Tap next
3. Select target app
4. Next
5. Schedule It(now)
6. Next
7. Additional options
8. Then review
9. Then publish

