LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK

MODUL XI

DATA STORAGE (BAGIAN II)

Disusun Oleh :

Aditya prabu mukti / 2211104037 SE-06-02

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

2024

GUIDED

1. LOGIC

Sebelum kita pergi ke tampilan flutter project, Langkah yang pertama kita lakukan adalah mengatur filefile services untuk notifikasi Firebase dan lain-lain

A. PUBSPEC.YAML

Sourcecode

```
name: pertemuan11
```

```
description: "A new Flutter project."
# The following line prevents the package from being accidentally published to
# pub.dev using `flutter pub publish`. This is preferred for private packages.
publish_to: 'none' # Remove this line if you wish to publish to pub.dev

# The following defines the version and build number for your application.
# A version number is three numbers separated by dots, like 1.2.43
# followed by an optional build number separated by a +.
# Both the version and the builder number may be overridden in flutter
# build by specifying --build-name and --build-number, respectively.
# In Android, build-name is used as versionName while build-number used as versionCode.
# Read more about Android versioning at https://developer.android.com/studio/publish/versioning
# In iOS, build-name is used as CFBundleShortVersionString while build-number is used as
CFBundleVersion.
# Read more about iOS versioning at
#
https://developer.apple.com/library/archive/documentation/General/Reference/InfoPlistKeyReference/Article
s/CoreFoundationKeys.html
# In Windows, build-name is used as the major, minor, and patch parts
# of the product and file versions while build-number is used as the build suffix.
version: 1.0.0+1

environment:
  sdk: ^3.5.3

# Dependencies specify other packages that your package needs in order to work.
# To automatically upgrade your package dependencies to the latest versions
# consider running `flutter pub upgrade --major-versions`. Alternatively,
# dependencies can be manually updated by changing the version numbers below to
# the latest version available on pub.dev. To see which dependencies have newer
# versions available, run `flutter pub outdated`.
dependencies:
  flutter:
    sdk: flutter

  # The following adds the Cupertino Icons font to your application.
  # Use with the CupertinoIcons class for iOS style icons.
  cupertino_icons: ^1.0.8
  firebase_core: ^3.8.0
  firebase_messaging: ^15.1.5
  flutter_local_notifications: ^18.0.1

dev_dependencies:
  flutter_test:
    sdk: flutter

  # The "flutter_lints" package below contains a set of recommended lints to
  # encourage good coding practices. The lint set provided by the package is
  # activated in the `analysis_options.yaml` file located at the root of your
  # package. See that file for information about deactivating specific lint
  # rules and activating additional ones.
  flutter_lints: ^4.0.0

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspec

# The following section is specific to Flutter packages.
```

```
flutter:

  # The following line ensures that the Material Icons font is
  # included with your application, so that you can use the icons in
  # the material Icons class.
  uses-material-design: true

  # To add assets to your application, add an assets section, like this:
  # assets:
  #   - images/a_dot_burr.jpeg
  #   - images/a_dot_ham.jpeg

  # An image asset can refer to one or more resolution-specific "variants", see
  # https://flutter.dev/to/resolution-aware-images

  # For details regarding adding assets from package dependencies, see
  # https://flutter.dev/to/asset-from-package

  # To add custom fonts to your application, add a fonts section here,
  # in this "flutter" section. Each entry in this list should have a
  # "family" key with the font family name, and a "fonts" key with a
  # list giving the asset and other descriptors for the font. For
  # example:
  # fonts:
  #   - family: Schyler
  #     fonts:
  #       - asset: fonts/Schyler-Regular.ttf
  #       - asset: fonts/Schyler-Italic.ttf
  #         style: italic
  #   - family: Trajan Pro
  #     fonts:
  #       - asset: fonts/TrajanPro.ttf
  #       - asset: fonts/TrajanPro_Bold.ttf
  #         weight: 700
  #
  # For details regarding fonts from package dependencies,
  # see https://flutter.dev/to/font-from-package
```

Deskripsi Program

File pubspec.yaml ini mengatur konfigurasi proyek Flutter bernama pertemuan11. Proyek ini dideskripsikan sebagai aplikasi Flutter baru yang tidak akan dipublikasikan ke pub.dev. Versi aplikasi ditetapkan sebagai 1.0.0+1, yang menunjukkan versi mayor 1, patch 0, dan build ke-1. Proyek ini menggunakan SDK Dart versi 3.5.3 atau yang lebih baru. Pada bagian dependencies, proyek ini mengandalkan beberapa paket, termasuk flutter sebagai kerangka kerja utama, cupertino_icons untuk ikon bergaya iOS, serta firebase_core, firebase_messaging, dan flutter_local_notifications untuk integrasi dengan Firebase dan pengelolaan notifikasi lokal. Selain itu, bagian dev_dependencies mencakup flutter_test untuk mendukung pengujian aplikasi dan flutter_lints untuk memastikan penerapan praktik pengkodean yang baik.

B. BUILD.GRADLE (ANDROID/APP)

Kita akan melakukan konfigurasi untuk build.gradle pada Android/app

Sourcecode

```
plugins {
    id "com.android.application"
    id "kotlin-android"
    // The Flutter Gradle Plugin must be applied after the Android and Kotlin Gradle plugins.
    id "dev.flutter.flutter-gradle-plugin"
    id 'com.google.gms.google-services'
}

android {
    namespace = "com.example.pertemuan11"
    compileSdk = flutter.compileSdkVersion
    ndkVersion = flutter.ndkVersion

    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_1_8
        targetCompatibility = JavaVersion.VERSION_1_8
    }

    kotlinOptions {
        jvmTarget = JavaVersion.VERSION_1_8
    }

    defaultConfig {
        // TODO: Specify your own unique Application ID (https://developer.android.com/studio/build/application-id.html).
        applicationId = "com.example.pertemuan11"
        // You can update the following values to match your application needs.
        // For more information, see: https://flutter.dev/to/review-gradle-config.
        minSdkVersion 21
        targetSdkVersion 33
        versionCode = flutter.versionCode
        versionName = flutter.versionName
    }

    buildTypes {
        release {
            // TODO: Add your own signing config for the release build.
            // Signing with the debug keys for now, so `flutter run --release` works.
            signingConfig = signingConfigs.debug
        }
    }
}

flutter {
```

```
    source = "../.."
}

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:33.6.0')

  // TODO: Add the dependencies for Firebase products you want to use
  // When using the BoM, don't specify versions in Firebase dependencies
  implementation 'com.google.firebase:firebase-analytics'

  // Add the dependencies for any other desired Firebase products
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

Deskripsi Program

File build.gradle ini mengatur konfigurasi proyek Android dalam aplikasi Flutter, mencakup pengaturan untuk plugin Android, Kotlin, dan Flutter, sekaligus integrasi dengan Firebase. File ini menetapkan versi SDK, ID aplikasi, dan konfigurasi build. Selain itu, Firebase BoM digunakan untuk secara otomatis mengelola versi layanan Firebase, termasuk pustaka seperti **firebase-analytics**. Konfigurasi ini dirancang untuk mendukung pengembangan aplikasi Flutter yang terintegrasi dengan layanan Firebase, memastikan kompatibilitas dan efisiensi dalam pengelolaan dependensi.

C. GOOGLE-SERVICES.JSON

Sourcecode

```
{
  "project_info": {
    "project_number": "877945545798",
    "project_id": "pertemuan11-8719e",
    "storage_bucket": "pertemuan11-8719e.firebasestorage.app"
  },
  "client": [
    {
      "client_info": {
        "mobilesdk_app_id": "1:877945545798:android:0983925a3cdd3c2c481647",
        "android_client_info": {
          "package_name": "com.example.pertemuan11"
        }
      },
      "oauth_client": [],
      "api_key": [
        {
          "current_key": "AIzaSyAENAdMbEJPQELToE09bfMW9A0kuBw_jVI"
        }
      ],
      "services": {
```

```
      "appinvite_service": {
        "other_platform_oauth_client": []
      }
    }
  }
 ],
 "configuration_version": "1"
}
```

Deskripsi Program

Proyek memiliki nomor proyek (877945545798), ID proyek (pertemuan11-8719e), dan bucket penyimpanan di Firebase Storage (pertemuan11-8719e.firebasestorage.app). Pada bagian client, terdapat informasi tentang aplikasi Android yang terhubung ke Firebase. Aplikasi ini memiliki mobilesdk_app_id berupa 1:877945545798:android:0983925a3cdd3c2c481647 dan nama paket aplikasi (com.example.pertemuan11).

Bagian api_key menyertakan kunci API (AIzaSyAENAdMbEJPQELToE09bfMW9A0kuBw_jVI) yang digunakan untuk mengakses layanan Firebase. Dalam bagian services, terdapat pengaturan untuk layanan undangan aplikasi (appinvite_service) tanpa klien OAuth untuk platform lain. File ini menunjukkan versi konfigurasi Firebase sebagai 1, yang mendukung integrasi aplikasi Flutter dengan layanan Firebase.

D. BUILD.GRADLE (ANDROID)

Sourcecode

```
buildscript {
    repositories {
        // Make sure that you have the following two repositories
        google() // Google's Maven repository
        mavenCentral() // Maven Central repository
    }
    dependencies {
        // Add the Maven coordinates and latest version of the plugin
        classpath 'com.google.gms:google-services:4.4.2'
    }
}
```

```
allprojects {
    repositories {
        google()
        mavenCentral()
    }
}

rootProject.buildDir = "../build"

subprojects {
    project.buildDir = "${rootProject.buildDir}/${project.name}"
}

subprojects {
    project.evaluationDependsOn(":app")
}

tasks.register("clean", Delete) {
    delete rootProject.buildDir
}
```

Deskripsi Program

File build.gradle ini mengatur konfigurasi repositori dengan menambahkan **Google** dan **Maven Central** sebagai sumber dependensi, serta menyertakan plugin **google-services** untuk mengintegrasikan Firebase ke dalam proyek. File ini juga menetapkan lokasi direktori build untuk proyek utama dan semua subproyek, memastikan bahwa subproyek **:app** dievaluasi terlebih dahulu. Selain itu, terdapat definisi tugas **clean**, yang berfungsi untuk menghapus direktori build proyek secara menyeluruh. Konfigurasi ini dirancang untuk mendukung pengembangan aplikasi Android yang terintegrasi dengan layanan Firebase.

2. TAMPILAN APLIKASI

Setelah kita melakukan konfigurasi untuk Firebase Notification, tidak lupa kita melakukan coding untuk tampilan aplikasinya

A. MY_NOTIFICATION_SCREEN.DART

Sourcecode

```
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';

// Variabel global
String? token;

Future<void> _firebaseMessagingBackgroundHandler(RemoteMessage message) async {
  await Firebase.initializeApp();
  print('Handling a background message: ${message.messageId}');
}

const AndroidNotificationChannel channel = AndroidNotificationChannel(
  'high_importance_channel', // ID Channel
  'High Importance Notifications', // Nama Channel
  description:
      'This channel is used for important notifications.', // Deskripsi Channel
  importance: Importance.high, // Prioritas
);

class MyNotificationScreen extends StatefulWidget {
  const MyNotificationScreen({super.key});

  @override
  State<MyNotificationScreen> createState() => _MyNotificationScreenState();
}

class _MyNotificationScreenState extends State<MyNotificationScreen> {
  final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin =
      FlutterLocalNotificationsPlugin();

  @override
  void initState() {
    super.initState();
// Membuat pengaturan inisialisasi notifikasi untuk Android
    var initializationSettingsAndroid =
        const AndroidInitializationSettings('@mipmap/ic_launcher');
    var initializationSettings =
        InitializationSettings(android: initializationSettingsAndroid);
    FlutterLocalNotificationsPlugin().initialize(initializationSettings);
// Mendengarkan pesan saat aplikasi aktif
    FirebaseMessaging.onMessageOpenedApp.listen((RemoteMessage message) {
      print("Notifikasi diklik: ${message.notification?.title}");
      RemoteNotification? notification = message.notification;
      AndroidNotification? android = message.notification?.android;

      if (notification != null && android != null) {
        showDialog(
```

```
          context: context,
          builder: (_) => AlertDialog(
            title: Text(notification.title ?? "No Title"),
            content: Text(notification.body ?? "No Body"),
            actions: [
              TextButton(
                onPressed: () {
                  Navigator.of(context).pop();
                },
                child: const Text("OK"),
              ),
            ],
          ),
        );
      }
    });

// Memanggil metode untuk mengambil token FCM perangkat
    getToken();
  }

// Metode untuk mendapatkan token FCM
  void getToken() async {
    token = await FirebaseMessaging.instance
        .getToken(); // Mendapatkan token FCM perangkat
    print('FCM Token: $token'); // Menampilkan token di log
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("My Notification Screen"),
        backgroundColor: Colors.amber,
      ),
      body: const Center(
        child: Text("Halaman untuk menerima notifikasi"),
      ),
    );
  }
}
```

Deskripsi Program

Kode ini digunakan untuk menangani notifikasi yang diterima dari Firebase Cloud Messaging (FCM) dalam aplikasi Flutter. Di layar MyNotificationScreen, aplikasi menginisialisasi notifikasi lokal di Android menggunakan FlutterLocalNotificationsPlugin. Ketika aplikasi menerima pesan FCM, dan pengguna membuka aplikasi dari notifikasi tersebut, akan muncul dialog yang menunjukkan informasi lengkap tentang notifikasi, seperti judul, isi pesan, dan informasi pengirim.

Selain itu, aplikasi juga mengambil token FCM perangkat dengan menggunakan getToken, yang kemudian dicetak ke log. Notifikasi diatur dengan saluran prioritas tinggi menggunakan AndroidNotificationChannel, sehingga dapat langsung muncul meskipun aplikasi berjalan di latar depan. Fungsi FirebaseMessaging.onMessageOpenedApp digunakan untuk menangani interaksi pengguna ketika mereka mengklik notifikasi, memungkinkan aplikasi menampilkan detail notifikasi dengan cara yang lebih interaktif.

B. MAIN.DART

Sourcecode

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:pertemuan11/my_notivication_screen.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  FirebaseMessaging.onBackgroundMessage(_firebaseMessagingBackgroundHandler);
  await FlutterLocalNotificationsPlugin()
      .resolvePlatformSpecificImplementation<
          AndroidFlutterLocalNotificationsPlugin>()
      ?.createNotificationChannel(channel);
  await FirebaseMessaging.instance.setForegroundNotificationPresentationOptions(
    alert: true,
    badge: true,
    sound: true,
  );
  runApp(const MyApp());
}

String? token;
Future<void> _firebaseMessagingBackgroundHandler(RemoteMessage message) async {
  await Firebase.initializeApp();
  print('Handling a background message: ${message.messageId}');
}

const AndroidNotificationChannel channel = AndroidNotificationChannel(
  'high_importance_channel', // ID Channel
  'High Importance Notifications', // Nama Channel
  description:
      'This channel is used for important notifications.', // Deskripsi Channel
  importance: Importance.high, // Prioritas
```

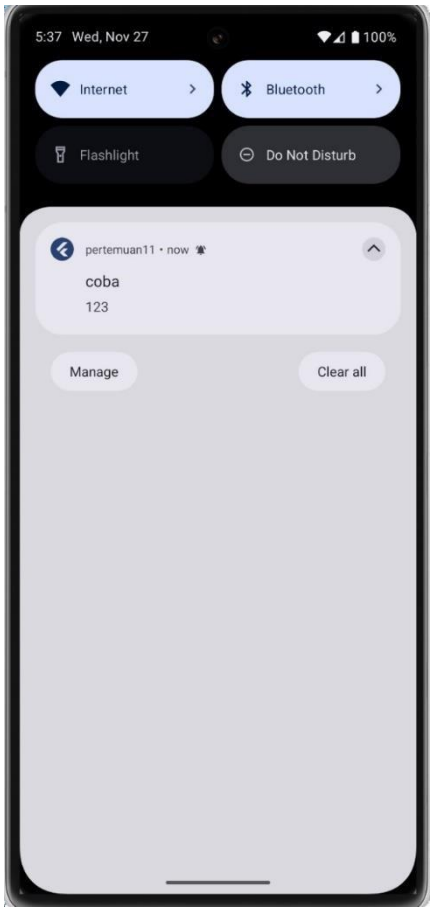
```
);

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // TRY THIS: Try running your application with "flutter run". You'll see
        // the application has a purple toolbar. Then, without quitting the app,
        // try changing the seedColor in the colorScheme below to Colors.green
        // and then invoke "hot reload" (save your changes or press the "hot
        // reload" button in a Flutter-supported IDE, or press "r" if you used
        // the command line to start the app).
        //
        // Notice that the counter didn't reset back to zero; the application
        // state is not lost during the reload. To reset the state, use hot
        // restart instead.
        //
        // This works for code too, not just values: Most code changes can be
        // tested with just a hot reload.
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyNotificationScreen(),
    );
  }
}
```

Deskripsi Program

Kode ini mengatur Firebase di aplikasi Flutter, mengonfigurasi notifikasi lokal dan Firebase Cloud Messaging (FCM). Untuk pesan FCM yang diterima di latar belakang, FirebaseMessaging.onBackgroundMessage menangani prosesnya, sedangkan untuk notifikasi lokal di latar depan, FlutterLocalNotificationsPlugin digunakan untuk menampilkan pesan penting. Aplikasi ini menggunakan tema Material3 dan menampilkan layar utama MyNotificationScreen untuk menerima dan menampilkan notifikasi.

Screenshoot Output

Tampilan aplikasi



Panel notifikasi

Alert dialog

Deskripsi Program

Gambar ini menggambarkan pengaturan Firebase dalam aplikasi Flutter untuk mengelola notifikasi menggunakan Firebase Cloud Messaging (FCM). Firebase diinisialisasi dan saluran notifikasi Android dikonfigurasi dengan prioritas tinggi agar pesan yang penting dapat ditampilkan. Selain itu, aplikasi juga diatur untuk menerima notifikasi baik saat berjalan di latar depan maupun latar belakang, serta menampilkan antarmuka pengguna (UI) yang responsif untuk menunjukkan notifikasi yang diterima.

UNGUIDED

Modifikasi Guided diatas bisa menampilkan Nama, Nim, Kelas, dan Prodi kalian ke dalam Notifikasi Flutter Cloud Messaging dan landing page notifikasinya.

A. MY_NOTIFICATION_SCREEN.DART

Sourcecode

```
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';

// Variabel global
String? token;

Future<void> _firebaseMessagingBackgroundHandler(RemoteMessage message) async {
  await Firebase.initializeApp();
  print('Handling a background message: ${message.messageId}');
}

const AndroidNotificationChannel channel = AndroidNotificationChannel(
  'high_importance_channel', // ID Channel
  'High Importance Notifications', // Nama Channel
  description:
      'This channel is used for important notifications.', // Deskripsi Channel
  importance: Importance.high, // Prioritas
);

class MyNotificationScreen extends StatefulWidget {
  const MyNotificationScreen({super.key});

  @override
  State<MyNotificationScreen> createState() => _MyNotificationScreenState();
}

class _MyNotificationScreenState extends State<MyNotificationScreen> {
  final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin =
      FlutterLocalNotificationsPlugin();

  @override
  void initState() {
    super.initState();

    // Membuat pengaturan inisialisasi notifikasi untuk Android
    var initializationSettingsAndroid =
        const AndroidInitializationSettings('@mipmap/ic_launcher');
    var initializationSettings =
        InitializationSettings(android: initializationSettingsAndroid);
    FlutterLocalNotificationsPlugin().initialize(initializationSettings);

    // Mendengarkan pesan saat aplikasi aktif
```

```
  FirebaseMessaging.onMessageOpenedApp.listen((RemoteMessage message) {
    print("Notifikasi diklik: ${message.notification?.title}");
    RemoteNotification? notification = message.notification;
    AndroidNotification? android = message.notification?.android;

    if (notification != null && android != null) {
      showDialog(
        context: context,
        builder: (_) => AlertDialog(
          title: const Text("Detail Notifikasi"),
          content: SingleChildScrollView(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text("Judul: ${notification.title ?? 'Tidak Ada Judul'}"),
                Text("Isi: ${notification.body ?? 'Tidak Ada Isi'}"),
                const SizedBox(height: 10),
                const Text("Informasi Pengirim:",
                    style: TextStyle(fontWeight: FontWeight.bold)),
                const Text("Nama: Aditya Prabu Mukti"),
                const Text("NIM: 2211104037"),
                const Text("Kelas: SE-06-02"),
                const Text("Prodi: S1 Software Engineering"),
              ],
            ),
          ),
          actions: [
            TextButton(
              onPressed: () {
                Navigator.of(context).pop();
              },
              child: const Text("Tutup"),
            ),
          ],
        ),
      );
    }
  });

  // Memanggil metode untuk mengambil token FCM perangkat
  getToken();
}

// Metode untuk mendapatkan token FCM
void getToken() async {
  token = await FirebaseMessaging.instance
      .getToken(); // Mendapatkan token FCM perangkat
  print('FCM Token: $token'); // Menampilkan token di log
}
```

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("My Notification Screen"),
        backgroundColor: Colors.amber,
      ),
      body: const Center(
        child: Text("Halaman untuk menerima notifikasi"),
      ),
    );
  }
}
```

B. MAIN.DART

Sourcecode

```
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:flutter_local_notifications/flutter_local_notifications.dart';
import 'package:firebase_messaging/firebase_messaging.dart';
import 'package:pertemuan11/my_notivication_screen.dart';

Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  FirebaseMessaging.onBackgroundMessage(_firebaseMessagingBackgroundHandler);
  await FlutterLocalNotificationsPlugin()
      .resolvePlatformSpecificImplementation<
          AndroidFlutterLocalNotificationsPlugin>()
      ?.createNotificationChannel(channel);
  await FirebaseMessaging.instance.setForegroundNotificationPresentationOptions(
    alert: true,
    badge: true,
    sound: true,
  );
  runApp(const MyApp());
}

String? token;
Future<void> _firebaseMessagingBackgroundHandler(RemoteMessage message) async {
  await Firebase.initializeApp();
  print('Handling a background message: ${message.messageId}');
}

const AndroidNotificationChannel channel = AndroidNotificationChannel(
  'high_importance_channel', // ID Channel
  'High Importance Notifications', // Nama Channel
  description:
```

```
    'This channel is used for important notifications.', // Deskripsi Channel
  importance: Importance.high, // Prioritas
);

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // TRY THIS: Try running your application with "flutter run". You'll see
        // the application has a purple toolbar. Then, without quitting the app,
        // try changing the seedColor in the colorScheme below to Colors.green
        // and then invoke "hot reload" (save your changes or press the "hot
        // reload" button in a Flutter-supported IDE, or press "r" if you used
        // the command line to start the app).
        //
        // Notice that the counter didn't reset back to zero; the application
        // state is not lost during the reload. To reset the state, use hot
        // restart instead.
        //
        // This works for code too, not just values: Most code changes can be
        // tested with just a hot reload.
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyNotificationScreen(),
    );
  }
}
```

Screenshoot output

Deskripsi Program

Program ini mengonfigurasi Firebase Cloud Messaging (FCM) dan Flutter Local Notifications untuk menangani notifikasi dalam aplikasi Flutter. Aplikasi ini menginisialisasi Firebase, mengatur channel notifikasi dengan prioritas tinggi, dan menampilkan notifikasi ketika aplikasi berada di latar depan atau latar belakang. Selain itu, aplikasi juga mengambil token perangkat FCM untuk mengirim notifikasi secara spesifik ke perangkat pengguna.