

BAB 2

LANDASAN TEORI

Pada bab ini berisikan tentang penjelasan apa saja yang akan digunakan dalam pembuatan aplikasi ini.

2.1 Rukun Tetangga

Menurut sejarahnya, Rukun Tetangga (RT) lahir melalui keputusan Menteri dalam negeri nomor 7 tahun 1983, yang pada dasarnya pendiriannya Lembaga tersebut ditujukan untuk membantu berbagai program pemerintah. RT adalah Lembaga kemasyarakatan, yaitu Lembaga yang dibentuk melalui musyawarah masyarakat setempat [3]. Adapun Fungsi RT sebagai berikut :

1. Pendataan kependudukan dan pelayanan administrasi pemerintahan lainnya.
2. Pemeliharaan ketertiban dan kerukunan hidup antar warga.
3. Pembuatan gagasan dalam pelaksanaan pembangunan dengan mengembangkan aspirasi dan swadaya murni masyarakat.
4. Penggerak swadaya gotong royong dan partisipasi masyarakat di wilayahnya.

Dari penjelasan tersebut diatas dapat disimpulkan tugas pengurus RT adalah sebagai berikut :

1. Menjaga kerukunan antar warga, melestarikan kegotong royongan dan kekeluargaan, dalam upaya menjaga ketertiban dan ketentraman masyarakat.
2. Menjalankan tugas pelayanan masyarakat.

Dengan demikian, dalam kesehariannya pengurus RT menangani berbagai urusan kemasyarakatan yang ada di wilayahnya. Adapun Berbagai urusan yang harus dilaksanakan meliputi :

1. Urusan ketertiban umum
 - 1.1 Kebersihan

- 1.2 Ketertiban
2. Urusan administrasi pelayanan masyarakat
 - 2.1 Administrasi Persuratan
 - 2.2 Administrasi Kependudukan
 - 2.3 Administrasi Keuangan
3. Urusan kesejahteraan masyarakat
 - 3.1 Kesehatan Masyarakat
 - 3.2 Kerukunan Umat beragama
4. Urusan pembangunan
 - 4.1 Swadaya masyarakat dan gotong-royong
 - 4.2 Pemeliharaan fasilitas dan lingkungan

Guna melaksanakan tugas dan fungsi tersebut diatas maka kepengurusan RT minimal akan terdiri dari :

1. Ketua
2. Sekretaris
3. Bendahara
4. Koordinator KKKLU

2.2 Android

Android adalah teknologi dibalik ponsel, tablet, jam, tv dan mobil yang kita gunakan, saat perangkat tidak hanya berfungsi, tapi juga membuat semuanya menjadi lebih mudah, android ada dibaliknya, sebagai contoh android membuat GPS anda dapat menghindari kemacetan, jam anda dapat mengirimkan teks, dan asistem anda dapat menjawab pertanyaan anda. Sistem operasi ini terdapat dalam 2,5 miliar perangkat aktif [4].

Android merupakan sistem operasi yang dikembangkan untuk perangkat mobile berbasis Linux. Pada awalnya sistem operasi ini dikembangkan oleh Android Inc. yang kemudian dibeli oleh Google pada tahun 2005 [5].

2.2.1 Sejarah Sistem Operasi Android

Di balik kesuksesan Android di masa sekarang, tentu saja sistem operasi ini memiliki latar belakang sejarah yang mungkin masih sedikit pengguna yang mengetahuinya. **Perkembangan sistem Android dimulai pada bulan September 2008**, di mana saat ini dilakukan peluncuran Android pertama versi 1.0.

Android 1.0 – 1.1

Untuk yang satu ini merupakan versi pertama dari Android, di mana saat itu dirilis pada tanggal 23 September 2008. Sebenarnya **Android 1.0** dulu dinamakan dengan Astro, namun karena terdapat kendala yang terjadi saat itu. Pihak android memutuskan untuk tidak menggunakan nama ini secara komersial. Meski demikian, versi ini pernah disematkan pada ponsel jenis HTC Dream.

Kemudian pada 9 Februari 2009 disusul dengan versi terbarunya yaitu **Android 1.1**. Seperti sebelumnya, pada versi ini juga memiliki kendala yang sama yaitu hak cipta merk. Meski versi ini terlihat sangat mendasar, namun perangkat lunak ini telah menyertakan rangkaian aplikasi Google seperti [Gmail](#), Maps, Kalender, [YouTube](#), atau Android Market.

Android 1.5 : Cupcake

Pada **Android 1.5** ini lah awal penamaan versi Android lahir, di mana saat itu diberikan nama Cupcake. Versi ini menyediakan banyak penyempurnaan pada antarmuka Android, termasuk Keyboard di layar utama. Namun yang hal menonjol pada versi Cupcake ini adalah kerangka kerja **untuk widget aplikasi pihak ketiga dan perekaman video**.

Android 1.6 : Donut

Versi **Android 1.6 Donut** dirilis pada 15 September 2009 ini berfokus pada beberapa celah pada Android, misalnya kemampuan OS untuk beroperasi di berbagai ukuran dan resolusi layar yang berbeda. Selain itu terdapat peningkatan dalam segi UI yang lebih *user friendly*.

Android 2.0 – 2.1 : Eclair

Setelah beberapa minggu sejak peluncuran Android Donut, **Android 2.0** dirilis dengan nama Eclair. Pada versi ini terdapat penambahan navigasi belokan demi belokan dengan panduan suara dan info lalu lintas *real – time*, atau mungkin sekarang disebut dengan Google Maps. Selain Navigasi, Eclair juga menghadirkan wallpaper animasi ke Android serta fungsi *Speech to text*.

Android 2.2 : Froyo

Sekitar empat bulan berlalu saat **Android 2.1**, Google menghadirkan langsung Android 2.2 dengan inisial nama Froyo. Android Froyo menghadirkan beberapa fitur menarik, misalnya penambahan fitur **Voice Actions**. Di mana Anda dapat melakukan fungsi dasar seperti mendapatkan petunjuk arah dan membuat catatan dengan mengetuk ikon lalu mengucapkan perintah.

Android 2.3 : Gingerbread

Android Gingerbread rilis pada tanggal 6 Desember 2010. Pembaruan kali ini sedikit berbeda dengan sebelumnya, di mana terdapat fitur dukungan [format video](#) dan juga adanya kamera depan.

Android 3.0 – 3.2 : Honeycomb

Android 3.0 dengan nama Honeycomb hadir pada awal tahun 2011, perilisannya ini dikhususkan untuk tablet. Untuk pembaruan pada versi 3.1 dan 3.2 berikutnya masih tetap sama menjadi entitas tablet eksklusif. Menariknya dalam versi Honeycomb ini, **memperkenalkan UI yang dirancang secara dramatis untuk Android**. Yaitu memiliki desain “holografik” yang menggambarkan kesan luar angkasa dan menekankan penggunaan ruang layar untuk tablet secara maksimal.

Android 4.0 : Ice Cream Sandwich

Pada 19 Oktober 2011, **Android 4.0** dengan nama Ice Cream Sandwich dirilis. Pada versi ini terdapat penyempurnaan konsep visual yang diperkenalkan pada Honeycomb sebelumnya. Meski demikian ICS menghilangkan sebagian besar tampilan “holografik” namun tetap menggunakan warna biru sebagai sorotan di

seluruh sistem. Selain itu, ICS juga hadir dengan fitur penghapusan notifikasi dan aplikasi terbaru.

Android 4.1 – 4.3 : Jelly Bean

Android mengeluarkan versi **Jelly Bean** pada tahun 2012. Versi ini telah memperkenalkan fitur Google Now, di mana berfungsi untuk berbagai keperluan secara cepat (*voice assistant*). Selain itu, Jelly Bean juga menempatkan widget di layar kunci Anda.

Android 4.4 : KitKat

13 Oktober 2013, **Android versi KitKat** diluncurkan dengan membawa peningkatan dari versi sebelumnya. Yaitu dalam hal *user experience*, dengan menampilkan bilah status yang transparan dan tampilan yang lebih kontemporer.

Android 5.0 : Lollipop

Android Lollipop rilis pada 12 November 2014 dengan adanya perubahan pada User Interface yang dibekali Material Design. Bahkan konsep UI ini membawa tampilan baru yang meluas di semua versi Android, aplikasi, hingga produk Google lainnya. **Perubahan yang paling menonjol pada versi ini adalah adanya kontrol suara melalui perintah “OK, Google” dan mode prioritas untuk manajemen pemberitahuan yang lebih baik.**

Android 6.0 : Marshmallow

Lain halnya dengan versi **Marshmallow** yang diluncurkan pada 5 Oktober 2015. Di mana pada versi ini menghadirkan beberapa fitur terbaru, yaitu fitur pencarian layar “Now On Tap”, perizinan aplikasi, dukungan sidik jari dan USB-C.

Android 7.0 : Nougat

Di tahun 2016, Google merilis **Android Nougat** bersamaan dengan produk Google “Pixel”. Pada versi ini terdapat perubahan yang cukup besar, misalnya

tersedia fitur Multi Window, [Penghemat Data](#), Pintasan Aplikasi, dan yang paling penting adalah Google Asisten.

Android 8.0 : Oreo

Android 8.0 Oreo rilis pada 21 Agustus 2017, di mana terdapat dukungan fitur *multi-tasking*, mode gambar, opsi [nonaktifkan pemberitahuan](#), dan juga [peringatan notifikasi](#). *User Interface* yang digunakan pun diperbarui supaya lebih rapi, segar dan pastinya memudahkan pengguna dalam mengakses aplikasi.

Android 9.0 : Pie

6 Agustus 2018, Google merilis **Android 9.0** dengan nama Pie. Dibanding sebelumnya, versi ini menghadirkan banyak fitur di dalamnya. Salah satu di antaranya adalah sistem navigasi gerakan/ tombol *hybrid* dan beranda multi-fungsi. Selain itu, Pie juga menyertakan beberapa fitur produktivitas seperti sistem yang lebih canggih untuk manajemen daya, [kecerahan layar](#), [penghemat baterai android](#), dan juga peningkatan privasi serta keamanan.

Android 10.0

Kemudian untuk **Android versi 10** ini tidak lagi menggunakan penamaan seperti yang dilakukan sebelumnya. Perubahan yang mencolok pada versi ini adalah menghadirkan antarmuka yang sepenuhnya ditata ulang. Di lanjutkan dengan perbaikan lainnya pada versi sebelumnya, seperti **perizinan aplikasi yang diperbarui**. Apalagi pada versi ini menyertakan tema gelap di seluruh sistem.

Android 11.0

Android 11 merupakan versi terbaru dari Android yang diluncurkan pada bulan September 2020. Terdapat banyak sekali perubahan pada versi ini, mulai dari peningkatan privasi dari versi sebelumnya dan lainnya. Misalnya seperti kemampuan pengguna memberikan izin tertentu pada aplikasi, membatasi aplikasi untuk berinteraksi dengan penyimpanan lokal, penyempurnaan notifikasi, memperkenalkan pemutar media baru untuk [aplikasi pemutaran video](#) atau audio dalam satu ruang, dan masih banyak lagi.

2.3 Dart

Dart adalah bahasa yang dioptimalkan untuk klien untuk mengembangkan aplikasi cepat di platform apa pun. Tujuannya adalah untuk menawarkan bahasa pemrograman paling produktif untuk pengembangan multi-platform, dipasangkan dengan [platform runtime eksekusi yang fleksibel](#) untuk kerangka kerja aplikasi. Dart dirancang untuk lingkup teknis yang sangat cocok untuk pengembangan klien, memprioritaskan pengembangan dan pengalaman produksi berkualitas tinggi di berbagai macam target kompilasi (web, seluler, dan desktop). Dart juga membentuk dasar Flutter. Dart menyediakan bahasa dan waktu proses yang mendukung aplikasi Flutter, tetapi Dart juga mendukung banyak tugas pengembang inti seperti memformat, menganalisis, dan menguji kode [6]. Berikut adalah gambar contoh implementasi Bahasa pemrograman Dart.

```
void main() {  
  print('Hello, World!');  
}
```

Gambar 2. 1 Contoh Dart

Setiap aplikasi memiliki main() function. Untuk menampilkan text di konsol, kita bisa menggunakan fungsi print() seperti gambar contoh diatas. Adapun penggunaan variabel seperti gambar berikut.

```
var name = 'Voyager I';  
var year = 1977;  
var antennaDiameter = 3.7;  
var flybyObjects = ['Jupiter', 'Saturn', 'Uranus', 'Neptune'];  
var image = {  
  'tags': ['saturn'],  
  'url': '//path/to/saturn.jpg'  
};
```

Gambar 2. 2 Contoh Dart Variabel

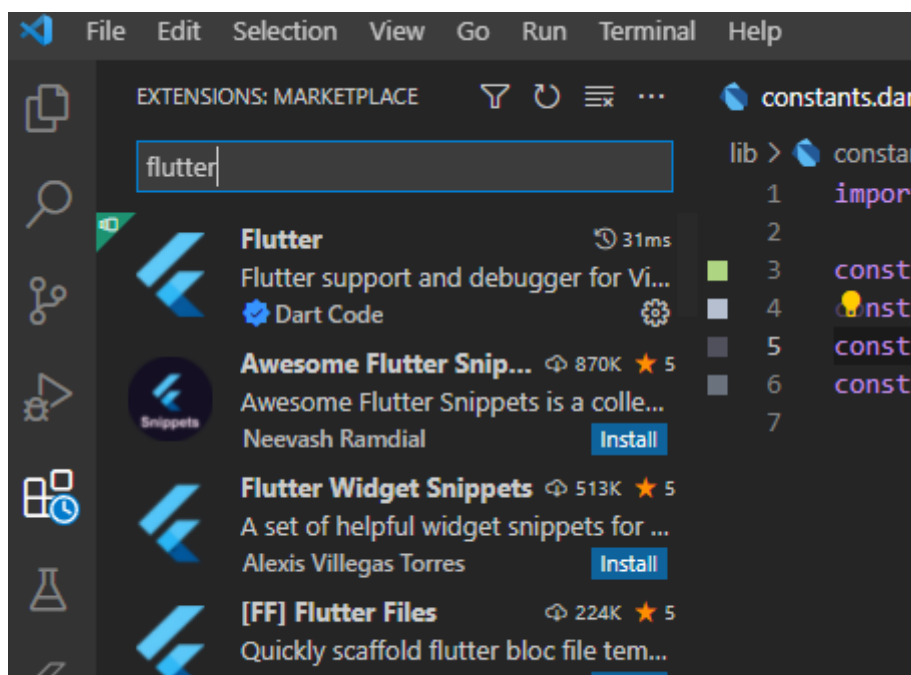
2.4 Flutter

Flutter adalah sebuah [framework aplikasi mobil sumber terbuka](#) yang diciptakan oleh [Google](#). Flutter digunakan dalam pengembangan aplikasi untuk sistem operasi [Android](#), [iOS](#), [Windows](#), [Linux](#), [MacOS](#), serta menjadi metode utama untuk membuat aplikasi [Google Funcsia](#). Flutter juga mendukung untuk pengembangan aplikasi berbasis web [7]. Ada beberapa peralatan yang harus disiapkan untuk membuat aplikasi dengan flutter, berikut peralatan yang harus disiapkan :

1. Teks Editor (Contoh : Visual studio code)
2. extension flutter
3. extension dart
4. Android SDK

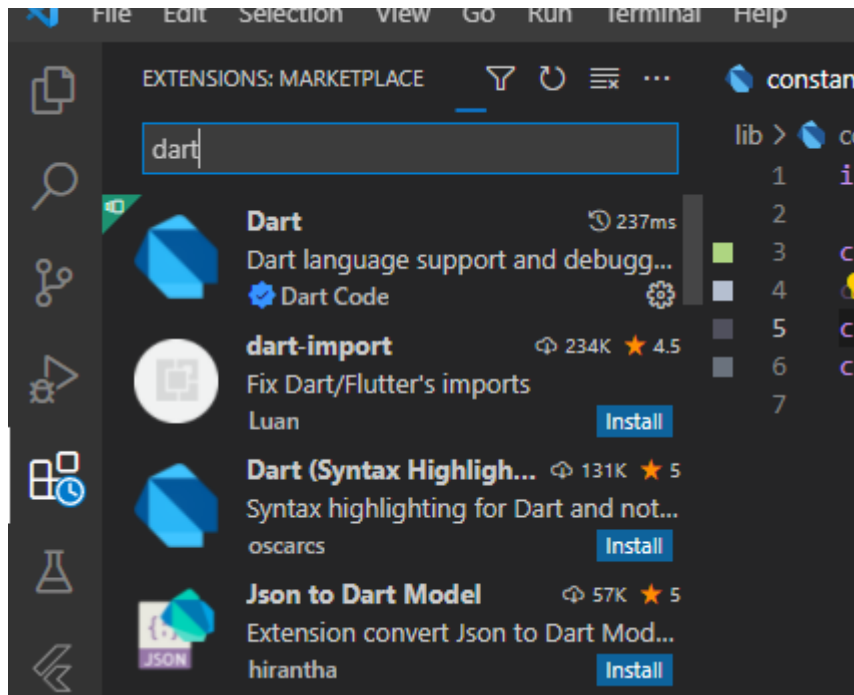
2.4.1 Cara menggunakan Flutter dengan VS code

Pertama silahkan download dan install teks editor visual studio code, Selanjutnya silahkan install beberapa extension Flutter yang dibutuhkan. Masuk ke manu extension, lalu ketik flutter.



Gambar 2. 3 Extension Flutter

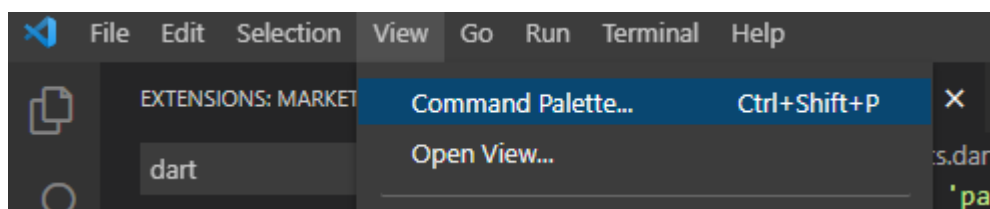
Setelah menginstall extension flutter,Langkah selanjutnya menginstall extension dart,seperti Langkah sebelumnya.



Gambar 2. 4 Extension Dart

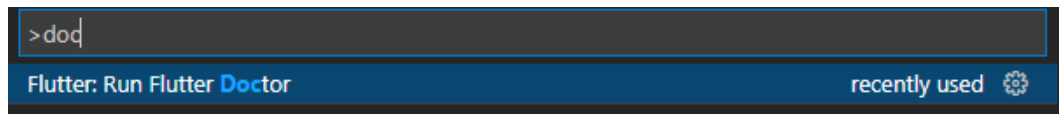
Setelah 'Flutter' dan 'Dart' sudah terinstall silahkan reload atau buka ulang Vs code, setelah menginstall extension tersebut kita menggunakan beberapa perintah seperti dibawah ini :

Tekan menu *View* lalu pilih *Command Palette*



Gambar 2. 5 Command Pallette

Lalu ketik flutter: Run Flutter Doctor



Gambar 2. 6 Run Flutter Doctor

Pastikan saat Run Flutter Doctor terhubung ke Internet. Setelah Run Flutter Doctor tunggu sampai proses pengecekan selesai seperti pada gambar dibawah:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
-----
• Dart version 2.4.0

[✓] Android toolchain - develop for Android devices (Android SDK version 28.0.3)
    • Android SDK at C:\Users\Dell\AppData\Local\Android\Sdk
    • Android NDK location not configured (optional; useful for native profiling support)
    • Platform android-28, build-tools 28.0.3
    • ANDROID_HOME = C:\Users\Dell\AppData\Local\Android\Sdk
    • Java binary at: C:\Program Files\Android\Android Studio\jre\bin\java
    • Java version OpenJDK Runtime Environment (build 1.8.0_202-release-1483-b03)
    • All Android licenses accepted.

[✓] Android Studio (version 3.5)
    • Android Studio at C:\Program Files\Android\Android Studio
    • Flutter plugin version 39.0.3
    • Dart plugin version 191.8423
    • Java version OpenJDK Runtime Environment (build 1.8.0_202-release-1483-b03)

[✓] VS Code (version 1.38.0)
    • VS Code at C:\Users\Dell\AppData\Local\Programs\Microsoft VS Code
    • Flutter extension version 3.4.1

[!] Connected device
    ! No devices available

! Doctor found issues in 1 category.
exit code 0

```

Gambar 2. 7 Output selesai pengecekan

2.5 Firebase

Firebase adalah suatu layanan dari Google untuk memberikan kemudahan bahkan mempermudah para developer aplikasi dalam mengembangkan aplikasinya. Firebase alias BaaS (Backend as a Service) merupakan solusi yang ditawarkan oleh Google untuk mempercepat pekerjaan developer. Firebase Realtime Database adalah database yang di-host di cloud. Data disimpan sebagai JSON dan disinkronkan secara realtime ke setiap klien yang terhubung. Ketika Anda mem-build aplikasi lintas platform dengan SDK iOS, Android, dan JavaScript, semua

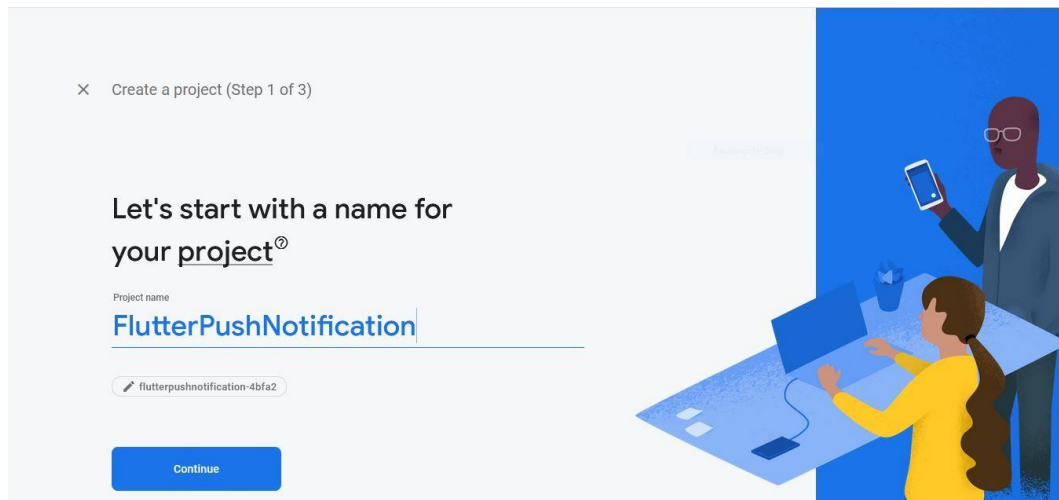
klien akan berbagi sebuah instance Realtime Database dan menerima update data terbaru secara otomatis. Dengan menggunakan Firebase, apps developer bisa fokus dalam mengembangkan aplikasi tanpa memberikan effort yang besar untuk urusan backend [8].

2.6 Firebase Cloud Messaging

Firebase Cloud Messaging (FCM) adalah solusi pengiriman pesan lintas platform yang memungkinkan Anda mengirim pesan secara andal tanpa biaya. Menggunakan FCM, Anda dapat memberi tahu aplikasi klien bahwa email baru atau data lain tersedia untuk disinkronkan. Anda dapat mengirim pesan notifikasi untuk mendorong interaksi ulang dan retensi pengguna. Untuk kasus penggunaan seperti pesan instan, sebuah pesan dapat mentransfer muatan hingga 4000 byte ke aplikasi klien [9].

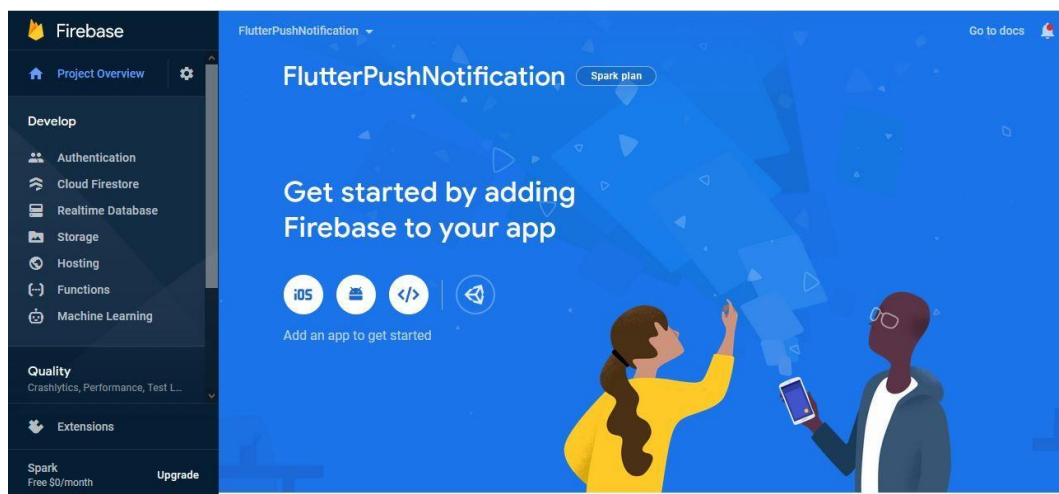
2.6.1 Instalasi FCM

Pada langkah ini, kita akan mengintegrasikan layanan Firebase dengan project Flutter kita. Pertama-tama, kita perlu membuat project Firebase. Panduan persiapan juga disediakan dalam [dokumentasi firebase](#) resmi untuk Flutter. Untuk membuat project Firebase, kita perlu masuk terlebih dahulu ke Firebase dan membuka konsol. Setelah terbuka kita cukup menekan 'create a project' untuk memulai project kita, kemudian akan halaman meminta memasukkan nama project seperti contoh gambar dibawah.



Gambar 2. 8 Membuat project

Selanjutnya setelah project dibuat maka akan muncul tampilan halaman seperti gambar berikut.



Gambar 2. 9 Tampilan halaman project firebase

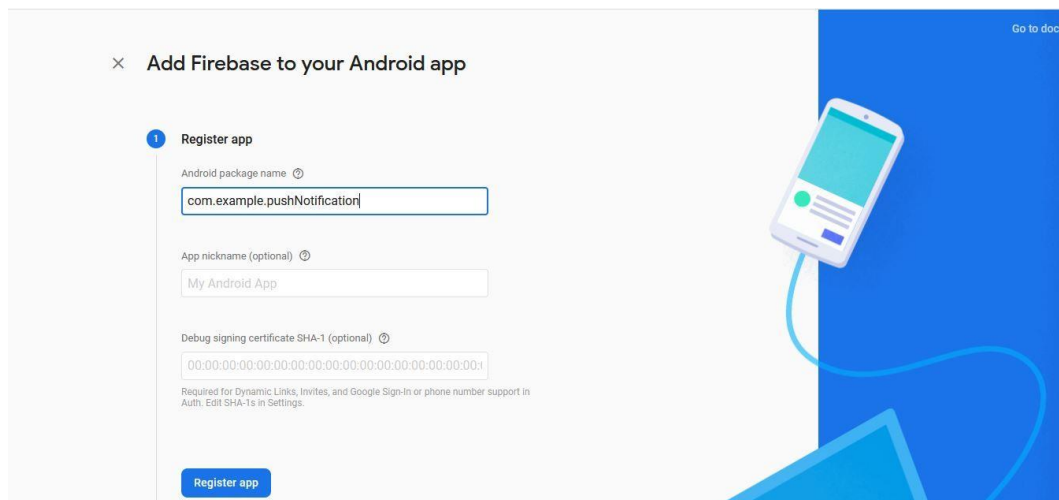
Di sini, kita akan menyiapkan Firebase untuk platform Android. Jadi kita perlu mengklik ikon Android yang ditampilkan pada tangkapan layar di atas. Ini akan mengarahkan kita ke antarmuka untuk mendaftarkan Firebase dengan proyek aplikasi Flutter kita.

Langkah selanjutnya memasukkan firebase kedalam aplikasi android yang ingin dibuat. Karena proses pendaftaran adalah khusus platform, kita akan mendaftarkan aplikasi kita untuk platform Android. Setelah mengklik ikon Android, kita akan diarahkan ke halaman yang menanyakan **nama package Android**. Untuk menambahkan nama package project Flutter, kita harus menemukannya terlebih dahulu. Nama package akan tersedia di file `./android/app/build.gradle` dari project Flutter seperti ini

```
com.example.pushNotification
```

Gambar 2. 10 package project

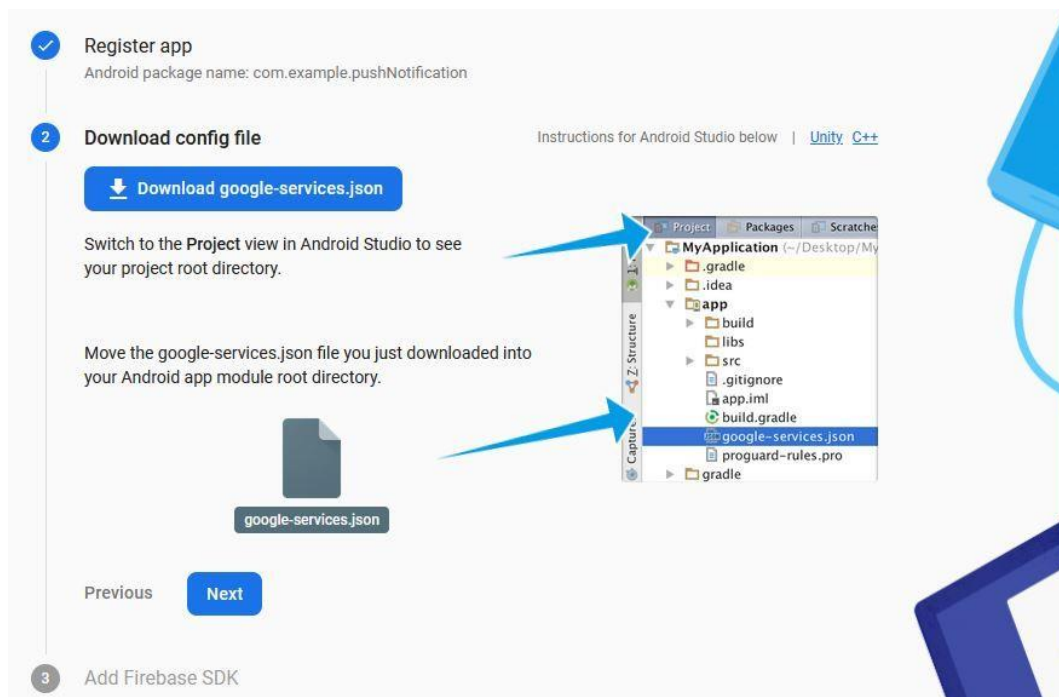
Kita cukup copy dan paste ke kolom android package name seperti gambar dibawah ini



The screenshot shows the 'Add Firebase to your Android app' interface. It has a title bar with a close button and the text 'Add Firebase to your Android app'. Below the title is a section '1 Register app'. There are three input fields: 'Android package name' with the value 'com.example.pushNotification', 'App nickname (optional)' with the value 'My Android App', and 'Debug signing certificate SHA-1 (optional)' which is empty. Below the third field is a note: 'Required for Dynamic Links, Invites, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.' At the bottom left is a blue 'Register app' button. On the right side, there is a blue background with a white smartphone and a laptop, and a 'Go to docs' link in the top right corner.

Gambar 2. 11 input android package name

Setelah itu, kita cukup mengklik tombol Register app. Ini akan membawa kita ke halaman kita mendapatkan file **google-services.json** yang akan menautkan aplikasi Flutter kita ke layanan Firebase Google. Kita perlu mengunduh file dan memindahkannya ke direktori `./android/app` dari project Flutter kita. Instruksi atau caranya ditunjukkan pada gambar berikut.



Gambar 2. 12 download google-services

Langkah selanjutnya adalah menambahkan konfigurasi firebase ke file project kita. Kita perlu menambahkan plugin google-serice ke file gradle kita.

Pertama di file gradle project kita di android/build.gradle, kita perlu menambahkan aturan untuk menyertakan plugin gradle layanan google. Kita perlu memeriksa apakah konfigurasi tersebut tersedia atau tidak, apabila tidak kita perlu menambahkan konfigurasi seperti gambar berikut

```

buildscript {
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
  }
  dependencies {
    ...
    // Add this line
    classpath 'com.google.gms:google-services:4.3.4'
  }
}

allprojects {
  ...
  repositories {
    // Check that you have the following line (if not, add it):
    google() // Google's Maven repository
    ...
  }
}

```

Sekarang di file gradle (android/app/build.gradle), kita perlu memasukkan plugin gradle layanan google. Untuk itu kita perlu menambahkan code berikut ke dalam android/app/build.gradle dari project kita.

```

// Add the following line:
**apply plugin: 'com.google.gms.google-services'** // Google Services plugin

android {
  // ...
}

```

Setelah semuanya disiapkan Langkah selanjutnya menjalankan command berikut agar konfigurasi package dapat dibuat.

```
flutter packages get
```

tunggu sampai proses konfigurasi selesai dan dengan itu kita telah berhasil mengintegrasikan firebase ke project kita.

Setelah selesai mengintegrasikan firebase ke project kita, selanjutnya memasukkan atau mengintegrasikan FCM dengan project. Pertama, kita perlu menambahkan dependensi firebase messaging ke file `./android/app/build.gardle`. Dalam file, kita perlu menambahkan dependensi berikut:

```
dependencies {
    implementation "org.jetbrains.kotlin:kotlin-stdlib-jdk7:$kotlin_version"
    implementation 'com.google.firebase:firebase-messaging:20.1.0'
}
```

Selanjutnya kita menambahkan suatu action dan kategori sebagai sebuah intent-filter didalam activity tag di file `./android/app/src/main/AndroidManifest.xml` file seperti gambar berikut.

```
<intent-filter>
    <action android:name="FLUTTER_NOTIFICATION_CLICK" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
```

Sekarang kita perlu membuat java file yang disebut `Application.java` di path `android/app/src/main/java/<app-organization-path>`, Lalu kita menambahkan code dari gambar berikut.

```
package io.flutter.plugins.pushNotification;

import io.flutter.app.FlutterApplication;
import io.flutter.plugin.common.PluginRegistry;
import io.flutter.plugin.common.PluginRegistry.PluginRegistrantCallback;
import io.flutter.plugins.GeneratedPluginRegistrant;
import io.flutter.plugins.firebaseessaging.FirebaseMessagingPlugin;
import io.flutter.plugins.firebaseessaging.FlutterFirebaseMessagingService;

public class Application extends FlutterApplication implements PluginRegistrantCallback {
    @Override
    public void onCreate() {
        super.onCreate();
        FlutterFirebaseMessagingService.setPluginRegistrant(this);
    }

    @Override
    public void registerWith(PluginRegistry registry) {
        FirebaseMessagingPlugin.registerWith(registry.registrarFor("io.flutter.plugins.firebaseessaging.FirebaseMessagingPlugin"));
    }
}
```


Setelah itu kita perlu assign application activity ke application tag yang ada di AndroidManifest.xml file seperti yang ditunjukkan pada potongan kode dibawah ini:

```
<application
    android:name=".Application"
```

Setelah semua dilakukan, persiapan firebase messaging plugin di native android code selesai. Sekarang kita pindah ke Flutter project kita.

Langkah yang akan kita lakukan sekarang adalah install firebase messaging package ke dalam project, kita menambahkan potongan code berikut di dependency ke dalam file pubspec.yaml.

```
cupertino_icons: ^1.0.2
flutter_local_notifications: ^9.5.3+1
workmanager: ^0.5.0
cloud_firestore: ^3.1.16
shared_preferences: ^2.0.15
crypt: ^4.2.1
firebase_storage: ^10.2.16
firebase_core: ^1.17.0
image_picker: ^0.8.5+3
cloud_functions: ^3.2.16
firebase_messaging: ^11.4.1
intl: ^0.17.0
pdf: ^3.8.1
path_provider: ^2.0.11
flutter_downloader: ^1.8.0+1
dio: ^4.0.6
url_launcher: ^6.1.3
http: ^0.13.4
```

Selanjutnya membuat UI screen yang simple saja di project kita atau tergantung tampilan yang diinginkan, berikut potongan code UI screen yang simple.

```
String messageTitle = "Empty";
String notificationAlert = "alert";

FirebaseMessaging _firebaseMessaging = FirebaseMessaging();
```

Variable message title akan menerima pesan notifikasi dan notification alert akan diberi action yang telah diselesaikan setelah notifikasi. Lalu kita terapkan code berikut kedalam project.

```
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(widget.title),
    ),
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          Text(
            notificationAlert,
          ),
          Text(
            messageTitle,
            style: Theme.of(context).textTheme.headline4,
          ),
        ],
      ),
    ),
  );
}
```

Setelah memasukkan source code diatas kita perlu menjalankan aplikasi flutter dengan command berikut di terminal projek :

```
flutter run
```

Kita akan mendapatkan hasil tampilan seperti gambar berikut :



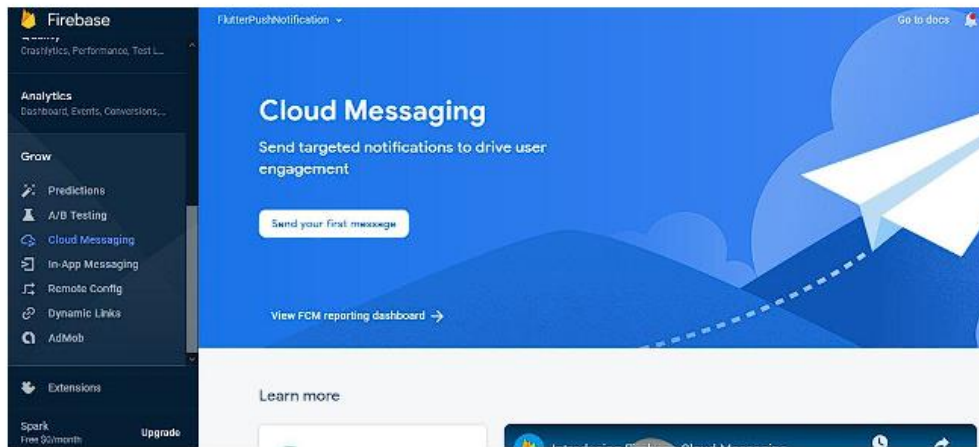
Untuk saat ini judul notifikasi masih kosong dan alert juga masih seperti yang ditentukan, kita perlu mengkonfigurasi kode untuk menerima notifikasi dan menggunakan notifikasi pesan agar bisa ditampilkan dilayar. Untuk itu kita perlu menambahkan potongan kode berikut kedalam `initstate()`.

```
@Override
void initState() {
  // TODO: implement initState
  super.initState();

  _firebaseMessaging.configure(
    onMessage: (message) async{
      setState(() {
        messageType = message["notification"]["title"];
        notificationAlert = "New Notification Alert";
      });
    },
    onResume: (message) async{
      setState(() {
        messageType = message["data"]["title"];
        notificationAlert = "Application opened from Notification";
      });
    },
  );
}
```

Di sini, kami telah menggunakan configure yang disediakan oleh `firebaseMessagingInstance` yang pada gilirannya menyediakan panggilan balik `onMessage` dan `onResume`. Callback ini memberikan notifikasi message sebagai parameter. Respons message akan menahan objek notifikasi sebagai objek peta.

Selanjutnya kita akan membuat pesan dari firebase cloud messaging console, pertama kita perlu membuka konsol cloud messaging di firebase seperti gambar berikut.



Disini kita dapat melihat opsi send your first message di layar, karena kita belum mengkonfigurasi pesan apapun sebelumnya. Kita perlu mengklik yang akan membawa kita ke halaman berikutnya seperti gambar dibawah ini.

1 Notification

Notification title ⓘ
Testing Message 1

Notification text
Testing Notification Message

Notification image (optional) ⓘ
Example: <https://yourapp.com/image.png>

Notification name (optional) ⓘ
Enter optional name

Device preview

This preview provides a general idea of how your message will appear on a mobile device. Actual message rendering will vary depending on the device. Test with a real device for actual results.

Send test message

Initial state Expanded view

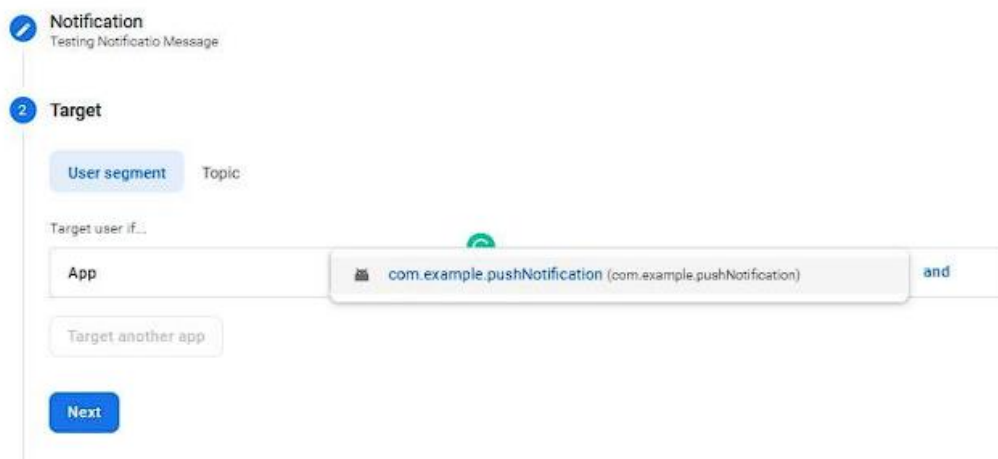
Android

iOS

Next

Di sini, kita bisa memasukkan judul, teks, gambar, dan nama notifikasi. Judul yang kita atur di sini akan diberikan sebagai judul di message objek pada callback yang kita atur sebelumnya di proyek Flutter.

Setelah mengatur bidang yang diperlukan, kita dapat mengklik 'Berikutnya' yang akan membawa kita ke halaman berikut:



Untuk step scheduling kita tetap default.



Terakhir, sebuah halaman di mana kita perlu memasukkan data khusus akan muncul di mana kita dapat mengatur titedan `click_action`. action klik ini dipicu setiap kali kita mengklik notifikasi yang muncul di bilah notifikasi perangkat.

Setelah mengklik pesan notifikasi dari bilah notifikasi, aplikasi akan terbuka dan `onResume` callback akan dipicu, pengaturan title seperti yang ditetapkan dalam data khusus pada gambar di bawah ini:

5 Additional options (optional)

All fields optional

Android Notification Channel ⓘ

Custom data ⓘ

title	Test Message
click_action	FLUTTER_NOTIFICATION_CLICK
Key	Value

Sound

Disabled ▾

Expires ⓘ

4 ▾ Weeks ▾

Sekarang kita siap untuk mengirimkan notifikasi ke perangkat.

2.7 Visual Studio Code

Visual studio code adalah source code editor yang ringan dan powerful yang dimana dijalankan di desktop dan tersedia di Windows, MacOS dan Linux. Mendukung untuk JavaScript, TypeScript dan Node.js dan memiliki ekosistem yang kaya akan extensions untuk Bahasa pemrograman dan runtime yang lainnya (seperti : C++, C#, Java, Python, PHP, Go, Dart) [10].

2.8 PDFkit

PDFKit adalah pembuatan dokumen PDF untuk Node dan browser yang membuat pembuatan dokumen menjadi kompleks, multi-halaman, dan dapat dicetak menjadi mudah. API mencakup kemampuan berantai, dan mencakup fungsi tingkat rendah serta abstraksi untuk fungsionalitas tingkat yang lebih tinggi. PDFKit API dirancang untuk mudah digunakan, sehingga menghasilkan dokumen yang kompleks seringkali sesederhana beberapa panggilan fungsi.

2.8.1 Cara install PDFkit

Pertama-tama buat folder dimanapun anda mau, buka cmd dan pindahkan ke direktori folder yang baru saja dibuat seperti gambar berikut :

```
E:\>cd E:\pdfkit  
E:\pdfkit>
```

Gambar 2. 13 Direktori folder PDFkit

Setelah itu meng-install PDFkit tersebut, instalasi PDFkit menggunakan *npm package manager* dengan mengetik perintah berikut :

```
E:\pdfkit>npm install pdfkit
```

Gambar 2. 14 Install PDFkit

Tunggu sampai proses instalasi selesai dan PDFkit bisa digunakan.

2.9.2 Cara menggunakan PDFkit

Setelah melakukan instalasi PDFkit, langkah selanjutnya ada membuat sebuah document menggunakan PDFkit. Pertama membuat document cukup mudah, cukup membutuhkan PDFkit module di source file javascript dan membuat PDFDocument seperti gambar berikut :

```
const PDFDocument = require('pdfkit');  
const doc = new PDFDocument;
```

Gambar 2. 15 Langkah pertama PDFkit

```
doc.pipe(fs.createWriteStream('output.pdf'));
```

Gambar 2. 16 Langkah kedua PDFkit

Selanjutnya melakukan menggunakan ukuran kertas,ada banyak pilihan ukuran kertas di PDFkit ini,berikut adalah ukuran kertas yang sering digunakan dan bisa dipilih di PDFkit.

EXECUTIVE (521.86 x 756.00)
 LEGAL (612.00 x 1008.00)
 LETTER (612.00 X 792.00)
 TABLOID (792.00 X 1224.00)

Gambar 2. 17 Ukuran Kertas PDFkit

PDFkit juga mendukung format ukuran kertas berikut ini :

4A0 (4767.89 x 6740.79)
 2A0 (3370.39 x 4767.87)
 FOLIO (612.00 X 936.00)

Gambar 2. 18 Ukuran Kertas PDFkit tambahan

Setelah memilih ukuran kertas yang diinginkan,masukkan format ukuran tersebut kedalam source code seperti contoh berikut.

```
// Passing size to the constructor
const doc = new PDFDocument({size: 'A7'});

// Passing size to the addPage function
doc.addPage({size: 'A7'});
```

Gambar 2. 19 Memilih ukuran Kertas

Ukuran kertas telah ditentukan,selanjutnya menuliskan sesuatu kedalam document yang akan dibuat, sebagai contohnya kita akan menulis hello world kedalam document pdf yang akan kita buat dengan memasukkan source code seperti contoh berikut.

```
doc.text('Hello world!')
```

Gambar 2. 20 Memasukan text kedalam PDFkit

Atau anda juga bisa mengatur posisi tulisan tersebut seperti gambar berikut.

```
doc.text('Hello world!', 100, 100)
```

Gambar 2. 21 memasukkan text kedalam PDFkit posisi yang diatur

Apabila document sudah dibuat sesuai yang diinginkan,kita ketikkan source code berikut untuk Finalize pdf file tersebut.

```
// Finalize PDF file
doc.end();
```

Gambar 2. 22 finalize PDFkit

Setelah selesai semua,save file tersebut dan jalankan command seperti gambar berikut.

```
E:\pdfkit>node index.js
```

Gambar 2. 23 Run file

Setelah menjalankan command tersebut maka akan muncul file pdf dengan

New folder	6/21/2022 4:08 PM	File folder	
node_modules	6/19/2022 12:26 PM	File folder	
index	6/19/2022 2:22 PM	JavaScript File	3 KB
jnt	10/29/2021 6:29 PM	PNG File	2 KB
logobekasi	6/19/2022 12:51 PM	PNG File	331 KB
output	6/19/2022 2:22 PM	Microsoft Edge P...	288 KB
package	6/21/2022 4:09 PM	Adobe After Effect...	1 KB
package-lock	6/21/2022 4:09 PM	Adobe After Effect...	53 KB

Gambar 2. 24 Hasil PDFkit

nama output.pdf di folder anda.

2.9 Payment Gateway

Payment gateway adalah istilah bahasa Inggris dari kata *payment* artinya pembayaran dan gateway berarti gerbang. Bila digabungkan artinya gerbang pembayaran adalah teknologi yang mampu membuat seluruh bisnis menerima transaksi pembayaran dari berbagai tempat dan waktu tak terbatas melalui pemanfaatan internet. Jadi pengertian payment gateway adalah alat pembayaran

suatu transaksi dalam layanan aplikasi e-commerce dengan fungsi mengotorisasi berbagai proses pembayaran baik perbankan, kartu kredit, transfer bank maupun secara langsung dari konsumen, atau yang lebih singkat dan jelasnya Payment gateway adalah pembayaran online yang fungsinya mendeskripsikan dan mengesahkan informasi pada sebuah transaksi sesuai dengan kebijakan yang telah diatur oleh para provider [11].

2.9.1 Manfaat atau kelebihan Payment Gateway

Payment gateway membawa dampak signifikan dalam kehidupan. Seluruh transaksi pembayaran berjalan secara efektif. Tak hanya itu, masih ada manfaat payment gateway antara lain:

1. Mempermudah transaksi online

Pertama, manfaat payment gateway adalah mempermudah transaksi online. Dengan adanya teknologi ini, Anda tidak perlu melakukan pembayaran secara offline melalui bank atau bertemu pihak penjual secara langsung. Karena proses transaksi telah otomatis dilakukan secara online.

2. Opsi Pembayaran Lebih Banyak

Opsi pembayaran lebih banyak merupakan manfaat payment gateway lainnya. Karena transaksi dilakukan secara online, maka penyediaan sarana pembayaran menjadi berkembang dan beragam. Anda bisa memilih sarana transaksi sesuai dengan opsi pembayaran yang dimiliki tanpa harus membuat dari awal.

3. Transaksi Lebih Cepat Diselesaikan

Manfaat penting dari payment gateway adalah transaksi lebih cepat diselesaikan. Anda tidak perlu menunggu beberapa hari untuk membeli suatu produk karena proses transaksi lama. Dengan adanya payment gateway, proses transaksi Anda segera selesai karena payment gateway terintegrasi langsung ke asosiasi penerbit/bank dan tempat transaksi barang.

4. Memungkinkan Transaksi Massal dalam Satu Waktu

Manfaat yang dirasakan oleh pelaku usaha dari payment gateway adalah transaksi dapat dilakukan secara massal dalam satu waktu. Hal ini

tentu membantu aktivitas bisnis lebih efisien. Misalnya seperti pengiriman gaji karyawan dilakukan secara efektif.

2.9.2 Payment Gateway Indonesia yang Populer untuk Transaksi Online

1. Finpay

Finpay merupakan salah satu payment gateway Indonesia terbaik dan terpopuler. Payment gateway yang satu ini merupakan anak perusahaan Telkom yang telah melayani payment gateway sejak tahun 2006. Dua layanan unggulan Finpay adalah Finpay Link dan Finpay Invoice. Finpay telah bekerja sama dengan beberapa perusahaan seperti Blibli dan Garuda Indonesia. Dalam memaksimalkan pelayanannya, Finpay juga bekerjasama dengan beberapa bank dan gerai pembayaran seperti BCA, BRI, Mandiri, Permata Bank, Danamon Online, CIMB, OVO, Gopay, Kredivo, Alfamart, Indomart, dan lain sebagainya.

2. Midtrans

Midtrans merupakan payment gateway yang sebelumnya bernama Veritrans Indonesia. Layanan ini berasal dari Jepang, namun sudah menjadi bagian dari Gojek sejak tahun 2017. Midtrans telah dipercaya lebih dari 500 ribu merchant termasuk di antaranya Tokopedia, KitaBisa dan TaniHub. Dalam memberikan layanannya, Midtrans menerima pembayaran melalui website, chat, hingga SMS.

3. Doku

Doku juga menjadi salah satu payment gateway Indonesia terbaik. Payment gateway yang satu ini telah berdiri sejak tahun 2007. Doku telah digunakan oleh sebanyak 150 ribu mitra dan tiga juta pengguna. Beberapa mitra Doku antara lain Lazada, Garuda Indonesia, KFC, HnM, Pertamina hingga Unicef. Tak hanya perusahaan besar, Doku juga telah bermitra dengan banyak UMKM.

2.10 Midtrans

Midtrans adalah salah satu penyedia layanan payment gateway Indonesia yang membantu mempermudah proses pembayaran, payment gateway pun memiliki fungsi lain seperti online payment yang tersedia 24 metode untuk kemudahan pelanggan lakukan pembayaran online. Instore payment yang berguna untuk toko offline agar menerima pembayaran langsung di outlet dan payouts dengan kirim dana kemana saja dengan mitra bank midtrans.

Midtrans cocok untuk startups dan early business, enterprise dan lain-lain. Keunggulan midtrans adalah menerima semua metode pembayaran, Solusi yang membantu terima semua transaksi online dan offline dengan 24 metode pembayaran. Pelanggan dapat memilih untuk gunakan e-Money, kartu kredit/debit, e-Banking, atau tunai di gerai. Dengan menawarkan pilihan, Anda bisa terima transaksi dengan semua pelanggan dari mana pun.

Pengaturannya yang cepat dan mudah, Platform yang dirancang dengan fleksibilitas maksimal. Pilih integrasi yang cocok sesuai kebutuhan Anda. Instalasi bisa dilakukan oleh siapa saja dan kompatibel dengan platform apa saja, dari Shopify hingga iOS. Terima dan lakukan transaksi dalam satu hari [12].

2.10.1 Sejarah dan perjalanan Midtrans

2012 midtrans didirikan dengan nama veritrans dan di 2013 membuka kantor di Bandung dan launching debit langsung, lalu di tahun 2014 launching virtual account dan pembayaran secara langsung, tahun 2016 veritrans mengubah Namanya menjadi midtrans dan mencapai kurang lebih 1000 pelanggan dan 100 staff bersamaan dengan launching aegis(sistem pendeteksi anomali) dan payouts. Pertengahan tahun 2017 diakuisisi oleh perusahaan GOJEK dan lunching selly, aplikasi asisten admin dan logistic midtrans, 2018 sampai sekarang adalah launching metode pembayaran gopay dan Qris [12].

2.11 Kamera

Kamera memiliki fungsi untuk menangkap sesuatu dari apa yang terlihat oleh kamera tersebut, kemudian di cetak kedalam lembaran kertas atau pada saat

ini dapat di cetak pada gambar digital. Sir John Herschel mengungkapkan istilah fotografi berasal dari bahasa Yunani yaitu *phos* yang berarti cahaya dan *graphein* yang berarti menggambar, dimana arti tersebut dapat diartikan menjadi menggambar dengan cahaya.

Teknologi kamera terus berevolusi hingga pada saat ini kamera dapat menyatu dengan perangkat smartphone, sehingga kamera lebih praktis dan dapat digunakan kapan saja. Kamera pada smartphone dapat digunakan untuk mengabadikan apapun dan gambar yang diabadikan disimpan pada smartphone tersebut. Kamera pada smartphone juga terus dikembangkan sehingga resolusi pixel pada kamera dapat menyaingi resolusi kamera DSLR dengan kualitas hasil yang baik.

Kinerja hardware kamera yang disematkan pada smartphone semakin hari semakin tinggi dengan kualitas hasil gambar yang mampu di tangkap juga semakin memanjakan mata penggunanya, begitu pula pengaruhnya terhadap perekaman video. Video yang dihasilkan oleh smartphone juga mengalami peningkatan terhadap kualitas video yang dihasilkan, bahkan beberapa vendor terkenal berlomba-lomba dalam menghasilkan video berkualitas full High Definition dengan resolusi pixel yang tinggi.

2.12 Cloud Function

Cloud Functions for Firebase adalah framework tanpa server yang memungkinkan menjalankan kode backend secara otomatis sebagai respons terhadap peristiwa yang dipicu oleh fitur Firebase dan permintaan HTTPS. Kode JavaScript atau TypeScript yang disimpan di cloud Google dan berjalan di lingkungan yang terkelola. Kita tidak perlu mengelola atau menskalakan server kita sendiri.

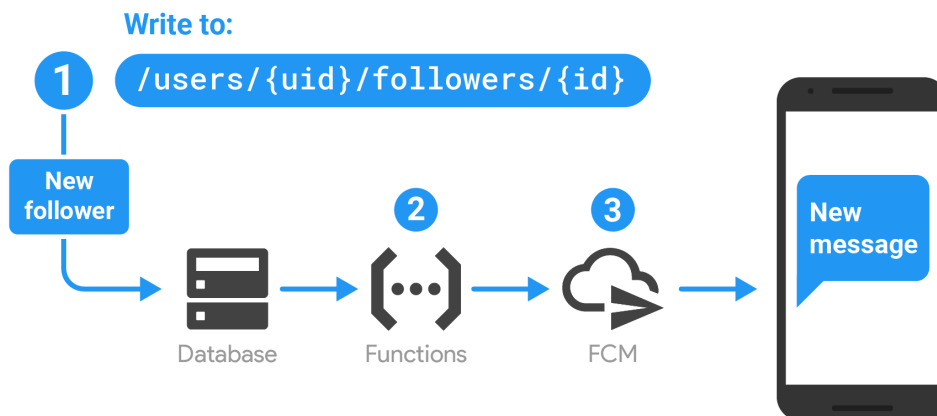
2.12.1 Cara Kerja Cloud Function

Setelah kita menulis dan men-deploy fungsi, server Google akan segera memulai pengelolaan fungsi. Kita dapat menjalankan fungsi secara langsung

dengan permintaan HTTP, atau dalam kasus fungsi latar belakang, server Google akan mendeteksi peristiwa dan menjalankan fungsi saat dipicu [13].

2.12.2 Contoh Kerja Cloud Function

Apa yang bisa dilakukan dengan Cloud Function, contohnya memberitahu pengguna apabila ada sesuatu yang menarik atau berubah. Developer dapat menggunakan Cloud Functions untuk mempertahankan interaksi pengguna dan memberikan informasi terbaru yang relevan tentang suatu aplikasi kepada pengguna. Misalnya, bayangkan suatu aplikasi yang memungkinkan penggunanya mengikuti aktivitas sesama pengguna aplikasi. Setiap kali pengguna menambahkan dirinya sendiri sebagai pengikut pengguna lain, peristiwa tulis akan terjadi di Realtime Database. Kemudian, peristiwa tulis ini dapat memicu fungsi untuk membuat notifikasi Firebase Cloud Messaging (FCM) yang akan memberi tahu pengguna terkait bahwa mereka memiliki pengikut baru.



Gambar 2. 25 Contoh Kerja Cloud Function

Fungsi ini dipicu ketika ada trigger atau peristiwa tulis ke jalur database realtime yang digunakan untuk menyimpan data pengikut, fungsi menuliskan pesan untuk dikirim melalui firebase cloud messaging dan FCM mengirimkan pesan notifikasi ke perangkat pengguna.

2.13 Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah sebuah bahasa yang telah menjadi standar untuk mendokumentasikan, menspesifikasikan, dan membangun perangkat lunak. Saat pertama kali diperkenalkan pada tahun 1997, saat ini UML telah berkembang menjadi sebuah bahasa pemodelan yang baku di dalam sebuah pengembangan perangkat lunak. UML digunakan dalam pengembangan perangkat lunak yang menggunakan pendekatan berorientasi objek. UML berkembang cukup pesat dan sebagiannya tergolong sebagai free software sehingga tersedia banyak pilihan seperti StarUML, ArgoUML, dan UML Designer [14].

UML menawarkan sebuah standar untuk perancangan model dari sebuah sistem. Model yang dibuat menggunakan UML dapat membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada sistem operasi, perangkat keras, dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML menggunakan class dan operation sebagai konsep dasarnya maka lebih cocok dalam pengembangan perangkat keras yang menggunakan pendekatan berorientasi objek.

2.13.1 Use Case Diagram

Use case diagram berisikan interaksi antara sekelompok proses dengan actor, fungsi apa saja yang ada di dalam sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Use case juga sangat penting untuk mengorganisasi dan memodelkan perilaku dari sesuatu sistem yang dibutuhkan serta diharapkan pengguna. Use case diagram memiliki beberapa komponen yaitu actor, dan use case, juga memiliki beberapa relasi yaitu association, generalization, dan dependency.

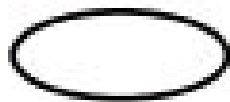
1. Actor



Gambar 2. 26 Simbol Aktor

Aktor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut, berinteraksi dengan use case tetapi tidak memiliki kontrol atas use case. Actor tersebut bisa merupakan manusia, sistem atau device.

2. Use Case



Gambar 2. 27 Simbol Use Case

Use case menggambarkan fungsionalitas yang disediakan oleh sistem sebagai unit-unit yang bertukar pesan antar unit dan actor, sehingga actor atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

3. Association



Gambar 2. 28 Simbol Association

Association merupakan penghubung antar element, mengindikasikan apa atau siapa yang meminta interaksi langsung dan bukan menggambarkan aliran data.

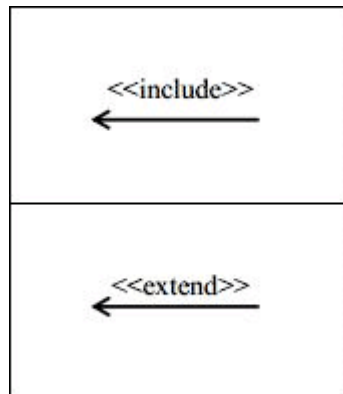
4. Generalization



Gambar 2. 29 Simbol Generalization

Mendefinisikan relasi antara dua aktor atau dua use case yang mana salah satunya mewariskan dan menambahkan atau override sifat dari yang lainnya.

5. Dependency

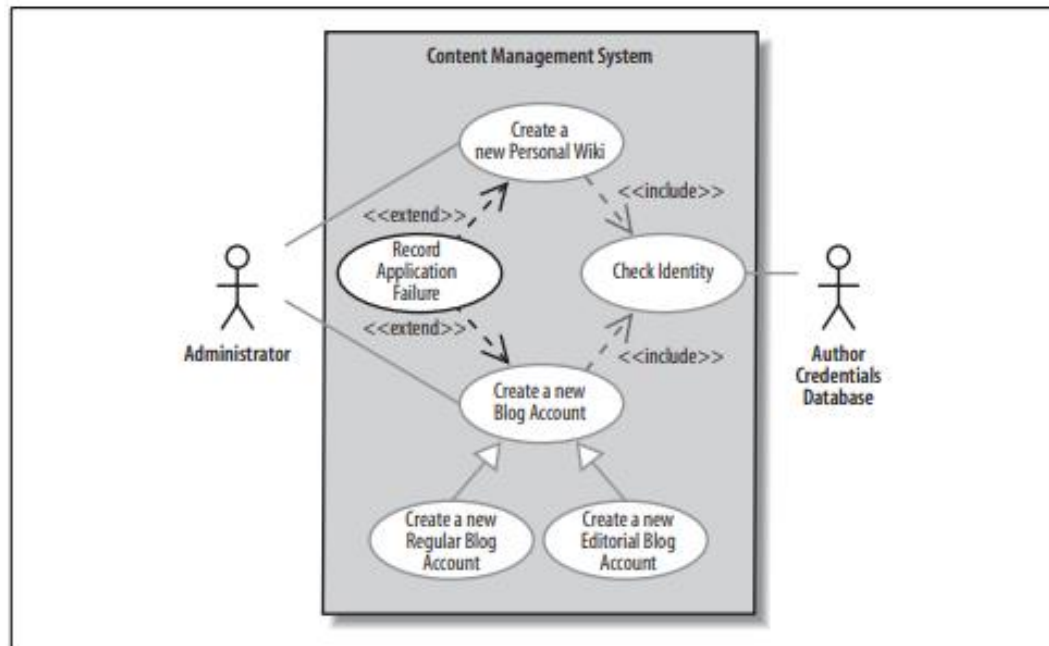


Gambar 2. 30 Simbol Dependency

Dependency terbagi menjadi 2 macam yaitu include yang berfungsi untuk mengindenfikasi hubungan antara 2 use case, dan extend merupakan perluasan dari use case lain jika kondisi atau syarat terpenuhi.

Untuk membuat sebuah Use Case Diagram tentunya membutuhkan *tools* atau aplikasi, Berikut adalah beberapa aplikasi yang bisa digunakan untuk membuat Use Case Diagram : Contoh Activity Diagram

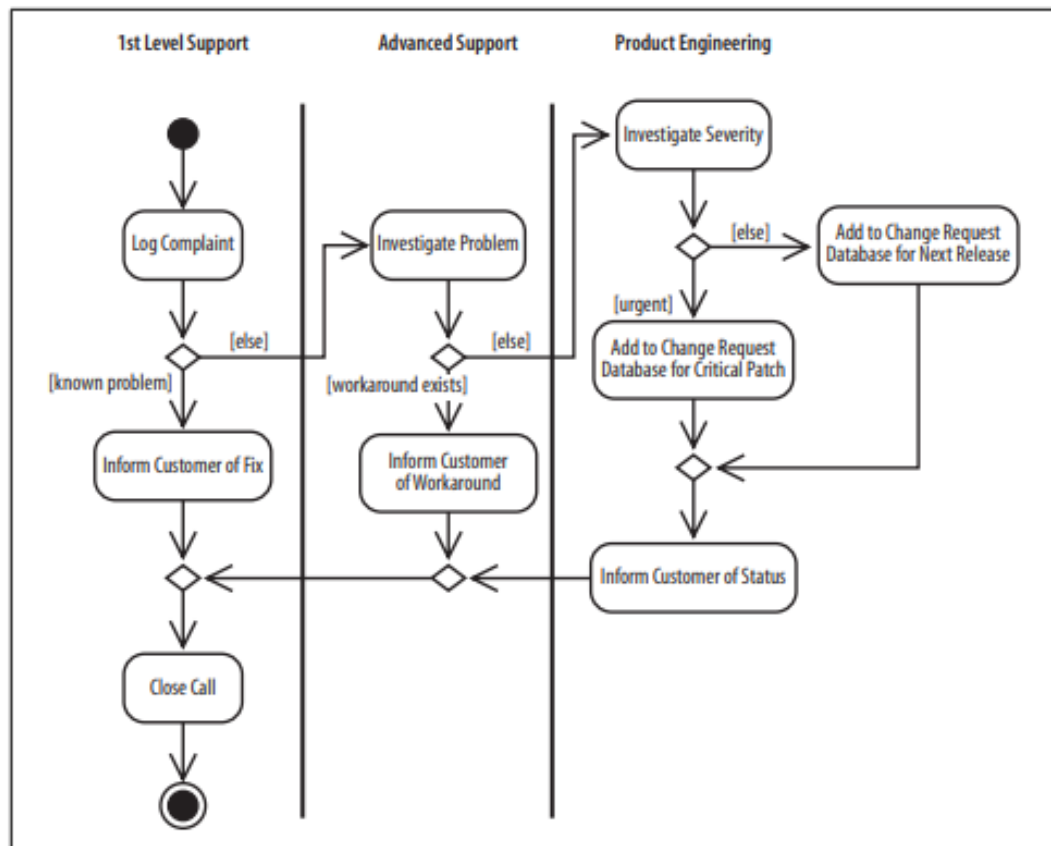
1. Draw.io
2. Visio



Gambar 2. 31 Contoh Usecase Diagram

2.13.2 Activity Diagram

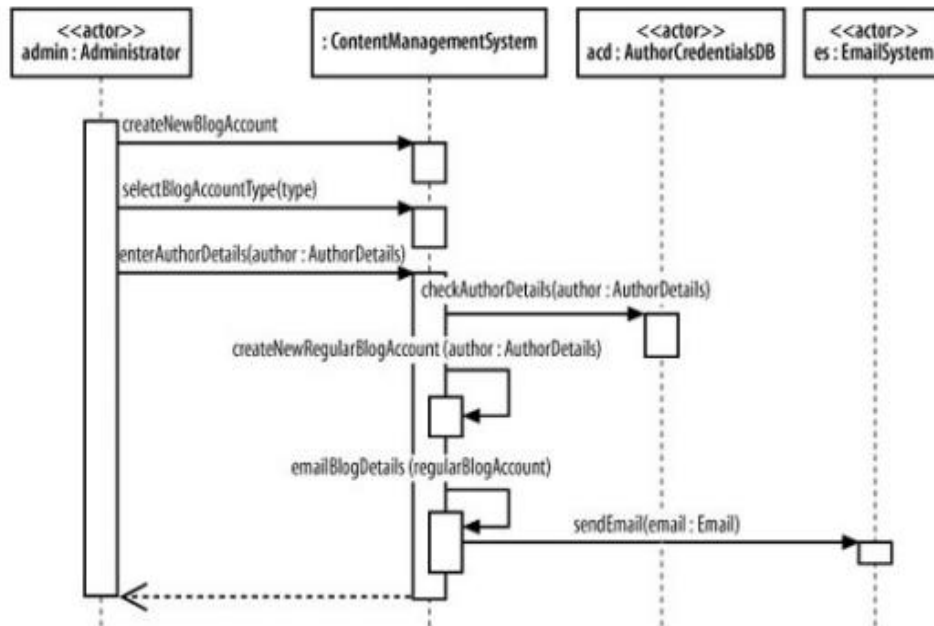
Activity diagram menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis. Activity diagram juga dapat menggambarkan proses lebih dari satu aksi dalam waktu yang bersamaan.



Gambar 2. 32 Contoh Activity Diagram

2.13.3 Sequence Diagram

Sequence diagram adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan use case diagram. Sequence diagram menggambarkan behavior internal sebuah sistem yang lebih menekankan pada penyampaian message dengan parameter waktu.

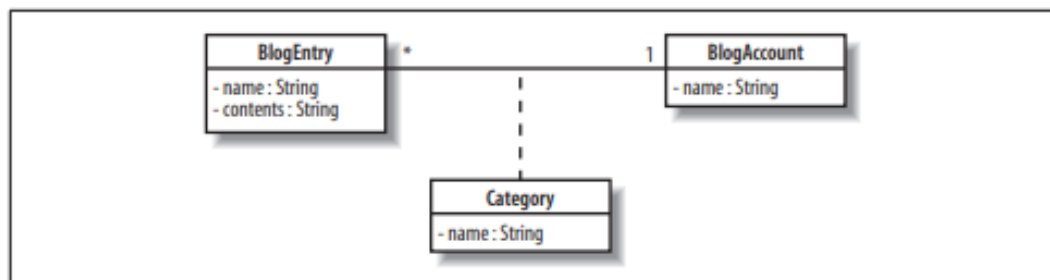


Gambar 2. 33 Contoh Sequence Diagram

2.13.4 Class Diagram

Class diagram merupakan hubungan antar kelas dan penjelasan dari tiap-tiap kelas tersebut di dalam model desain dari suatu sistem, yang juga memperlihatkan aturan dan tanggung jawab sistem yang menentukan perilaku sistem. Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan constraint yang berhubungan dengan objek yang dikoneksikan.

Class diagram secara khas meliputikelas, relasi association, generalitation maupun aggregation, attribute, operasi, dan visibility. Hubungan antar kelas mempunyai keterangan yang disebut sebagai Multiplicity dan Cardinality.



Gambar 2. 34 Contoh Class Diagram

