BAB 2
LANDASAN TEORI

1.1 Aplikasi

Aplikasi mengacu pada sebuah program komputer yang diciptakan untuk memenuhi kebutuhan dalam berbagai aktivitas seperti bisnis, permainan, layanan publik, periklanan, dan hampir semua proses kegiatan lainnya[10]. Perangkat lunak aplikasi merupakan bagian dari software komputer yang langsung memanfaatkan kemampuan komputer untuk melakukan tugas yang diinginkan pengguna, berbeda dengan perangkat lunak sistem yang lebih berfokus pada integrasi kemampuan komputer tanpa langsung menjalankan tugas yang bermanfaat bagi pengguna. Contoh umum dari perangkat lunak aplikasi adalah program pengolah kata, *spreadsheet*, dan pemutar media.

1.2 Monitoring

Monitoring adalah langkah-langkah sistematis dan berkelanjutan dalam mengumpulkan serta menganalisis informasi berdasarkan indikator yang telah ditetapkan terkait dengan kegiatan program. Tujuannya adalah untuk memungkinkan dilakukannya tindakan koreksi guna meningkatkan kualitas program kegiatan tersebut di masa mendatang[11]. Proses pemantauan yang dilakukan dengan kesadaran akan informasi yang ingin diketahui. Monitoring pada tingkat yang tinggi dilakukan untuk memungkinkan pengukuran dari waktu ke waktu yang menunjukkan pergerakan menuju atau menjauhi tujuan yang ditetapkan. Informasi yang diberikan oleh monitoring mencakup status dan tren, yang digunakan dalam pengukuran dan evaluasi yang dilakukan secara berkala. Monitoring umumnya dilakukan untuk tujuan tertentu, seperti memeriksa proses dan objek, atau untuk mengevaluasi kemajuan menuju tujuan manajemen. Ini melibatkan tindakan untuk menjaga konsistensi dalam manajemen yang sedang berlangsung.

1.3 Anak

Menurut Kamus Besar Bahasa Indonesia (KBBI), anak mengacu pada keturunan atau individu yang masih berusia kecil. Dalam penggunaan sehari-hari, anak mengacu pada seseorang yang belum mencapai usia tertentu atau belum menikah, yang sering dijadikan sebagai panduan umum. Dari perspektif hukum,

definisi anak dalam hukum positif Indonesia adalah individu yang belum mencapai kedewasaan, berada di bawah umur, atau berada di bawah pengawasan seorang wali[12]. Anak atau kanak-kanak secara umum merujuk pada sesuatu yang lebih kecil, entitas yang belum mencapai dewasa, atau objek yang berada di bawah kendali objek lain. Namun, interpretasi ini bervariasi sesuai dengan disiplin ilmiah yang bersangkutan. Dalam konteks biologi, anak mengacu pada makhluk hidup yang belum mencapai tahap kematangan atau dewasa, terutama pada hewan yang belum siap kawin, dan kadang-kadang pada tumbuhan untuk merujuk pada pertumbuhan awal dari umbi atau rumpun tumbuhan besar. Dalam bidang psikologi, anak merujuk pada individu yang belum mencapai kedewasaan fisik dan mental, biasanya dari masa bayi hingga masa sekolah dasar atau remaja. Dalam silsilah keluarga, anak adalah generasi kedua setelah orang tua, dan dalam konteks hukum Indonesia, anak adalah mereka yang berusia di bawah 18 tahun, termasuk yang masih dalam kandungan. Secara organisasional, anak atau anak buah adalah mereka yang bekerja di bawah pimpinan, sedangkan dalam kelompok objek yang berbeda ukuran, anak adalah yang lebih kecil dari yang lainnya.

1.4 Android

Android adalah sistem operasi yang berjalan pada perangkat seluler seperti smartphone dan tablet dan dimiliki oleh Google. Google menciptakan sistem operasi Android bersama sejumlah produsen besar, termasuk HTC, Samsung, Intel, dan Qualcomm, untuk membentuk Open Handset Alliance (OHA)[13].

Kode sumber Android bersifat open source karena merupakan versi modifikasi dari sistem operasi Linux 2.6. Android dapat berfungsi pada perangkat seluler dengan spesifikasi terbatas karena terdiri dari berbagai program dan pustaka Java yang memanfaatkan arsitektur berorientasi objek Java dan Dalvik Virtual Machine (DVM).

1.4.1 Android SDK

Android SDK (*Software Development Kit*) adalah seperangkat alat pengembangan yang digunakan untuk mengembangkan aplikasi untuk platfotm Android. Ini merupakan kumpulan alat pengembangan perangkat lunak dalam satu paket yang dapat diinstal yang mencangkup debugger, pustaka, emulator handset berdasarkan QEMU, dokumentasi, kode contoh, dan tutorial.

SDK adalah bagian penting dari pengembangan Android untuk dipahami oleh pemula. Ini menyediakan pilihan alat yang diperlukan untuk membangun aplikasi Android dan memastikan prosesnya berjalan semulus mungkin. Platform SDK diperbarui secara berkala, dan penyempurnaan SDK Android berjalan seiring dengan pengembangan platform Android secara keseluruhan. SDK juga mendukung versi platform Android yang lebih lama jika pengembang ingin menargetkan aplikasi mereka di perangkat yang lebih lama. SDK tersedia untuk komputer yang menjalankan Linux, Windows, dan Mac OS X. Komponen Android SDK dapat diunduh secara terpisah[14].

1.4.2 Android Studio

Android Studio merupakan Integrated Development Environment (IDE) yang secara khusus dirancang untuk memfasilitasi pengembangan aplikasi Android. IDE ini menyediakan berbagai fitur yang mendukung seluruh proses pengembangan mulai dari pembuatan kode, debugging, hingga pengujian aplikasi. Dengan Android Studio, para pengembang dapat membuat antarmuka pengguna yang interaktif melalui XML, menulis logika aplikasi dengan bahasa pemrograman Java atau Kotlin, mengakses berbagai fitur perangkat seperti kamera dan lokasi, serta mengintegrasikan berbagai layanan dan API dari Google, termasuk Google Maps API, Firebase, dan lainnya. Selain itu, Android Studio juga dilengkapi dengan emulator yang memungkinkan para pengembang menguji aplikasi mereka dalam berbagai konfigurasi perangkat Android sebelum aplikasi diluncurkan ke perangkat fisik. Dengan fitur-fitur dan dukungan yang lengkap ini, Android Studio menjadi sebuah lingkungan pengembangan yang sangat efektif dan efisien bagi para pengembang aplikasi Android.

1.5 Java

Java adalah sebuah bahasa pemrograman yang bisa digunakan pada berbagai perangkat, termasuk ponsel. Dikembangkan oleh James Gosling di Sun Microsystems, yang kini menjadi bagian dari Oracle, Java dirilis pada tahun 1995. Bahasa ini mengadopsi banyak sintaks dari C dan C++, namun dengan model objek yang lebih sederhana dan dukungan rutin rendah tingkat yang minimal. Aplikasi-aplikasi berbasis Java biasanya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan di berbagai Mesin Virtual Java (JVM). Java secara luas digunakan dalam

pengembangan berbagai jenis aplikasi, termasuk aplikasi desktop, perangkat lunak server, aplikasi mobile (Android), dan pengembangan web (JavaServer Pages)[15].

1.6 GPS *(Global Positioning System)*

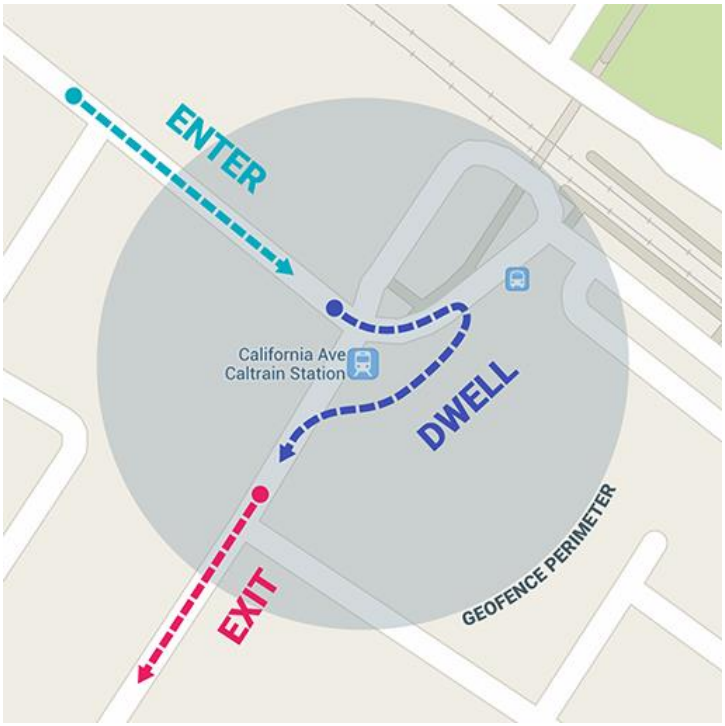
GPS *(Global Positioning System)* adalah sistem navigasi satelit yang digunakan untuk menentukan posisi dan waktu di seluruh dunia dengan akurasi tinggi. Sistem ini awalnya dikembangkan untuk keperluan militer oleh departemen Amerika Serikat. GPS bekerja dengan menggunakan jaringan satelit yang mengorbit bumi. Dua puluh emapat satelit aktif ditempatkan di orbit untuk memastikan adanya sinyal yang tersedia di seluruh dunia[16]. GPS Tracker atau GPS Tracking adalah sebuah teknologi AVL (*Automated Vehicle Locater*) yang memungkinkan pengguna untuk memantau secara real-time posisi kendaraan, armada, atau mobil. Teknologi ini menggabungkan GPS dan GSM untuk menentukan koordinat suatu objek dan menampilkan informasi tersebut dalam bentuk peta digital. Sinyal GPS yang disiarkan oleh satelit Global Positioning System digunakan untuk navigasi satelit. Dengan bantuan penerima di dekat permukaan bumi, orang dapat menentukan lokasi, waktu, dan kecepatan menggunakan informasi ini. Konstelasi satelit GPS dioperasikan oleh Skuadron Operasi Luar Angkasa ke-2 (2SOPS) Space Delta 8, Angkatan Luar Angkasa Amerika Serikat. Sinyal GPS mencakup sinyal jangkauan yang mengukur jarak ke satelit dan pesan navigasi, termasuk data ephemeris untuk menghitung posisi setiap satelit dan informasi tentang waktu dan status seluruh konstelasi satelit yang disebut almanak. Data lokasi akan dikirim melalui berbagai jenis jaringan seperti jaringan seluler GPRS (General Packet Radio Service), atau modem satelit yang sudah terintegrasi dalam perangkat yang sedang digunakan[17].

1.7 Geofencing

Geofencing merupakan parameter virtual yang digunakan untuk menetapkan batas wilayah geografis di dunia nyata. Parameter ini dapat berupa dinamis atau statis. Sebagai contoh, geofence dinamis dapat berupa radius sekitar atau titik lokasi tertentu, sedangkan geofence statis dapat mengacu pada batas-batas yang sudah ditetapkan sebelumnya seperti zona sekolah atau batas lingkungan. Penggunaan geofencing digital juga mungkin jika perangkat mendukung layanan berbasis lokasi (LBS). Ketika pengguna didalam atau diluar dari geofencing,

perangkat akan menerima pesan notifikasi yang berisi informasi tentang lokasi perangkat. Pemberitahuan geofence ini dapat dikirimkan ke ponsel untuk memicu tindakan-tindakan lainnya[18]. Penggunaan pembatasan wilayah menggabungkan lokasi saat ini pengguna dengan kedekatan lokasi tertentu. Untuk menandai lokasi, kita harus mengidentifikasi lintang dan bujur, sementara untuk menyesuaikan kedekatan, kita tambahkan radius. Lintang, bujur, dan radius ini membentuk area pembatasan wilayah, yang menciptakan area lingkaran di sekitar lokasi yang ditetapkan.

Berikut Geofencing dapat dilihat pada gambar 2.1:



Gambar 0.1 Geofencing

Sumber: https://developer.android.com/images/training/geofence.png

Dalam Gambar 2.1 di atas, terlihat sebuah target atau objek yang memasuki area yang telah ditentukan menggunakan geofence. Ketika target memasuki area atau keluar area tersebut, maka sistem akan memicu notifikasi yang berisi informasi tertentu yang dipicu oleh sistem[17]. Penggunaan geofencing dapat diterapkan dalam berbagai bidang terkait pemantauan target atau informasi yang spesifik.

1.8 Geofance Area

Selain menggunakan lingkaran sebagai pembatas geofencing, kita juga dapat menggunakan berbagai bentuk geometris lainnya seperti segitiga, segiempat, segilima, segienam, atau bentuk yang lebih kompleks seperti segibanyak yaitu polygon. Pembatas geofencing juga bisa berupa jalur terbuka atau bentuk bangunan yang kompleks seperti segibanyak. Setelah pembuatan pembatas geofencing sebagai pagar digital, objek yang bergerak dapat dipantau dengan lebih efektif[19].

Berikut salah satu sampel polygon bisa dilihat pada gambar 2.2:

Gambar 0.2 Sampel Polygon

Sumber: https://developers.google.com/static/maps/documentation/android-sdk/images/polygon-tutorial.png

Untuk memonitor pergerakan objek, diperlukan penggunaan modul GPS yang dapat memberikan data seperti longitude, latitude. Longitude mengindikasikan posisi objek pada garis bujur, sedangkan latitude menunjukkan posisinya pada garis lintang.

1.9 LBS (*Location Based Service*)

LBS (*Location Based Service*) merujuk pada layanan informasi yang memanfaatkan kemampuan untuk menentukan lokasi perangkat seluler, yang dapat diakses melalui jaringan seluler. Fungsinya adalah untuk membantu menemukan posisi seseorang atau suatu objek dengan menggunakan teknologi lokasi[20]. Istilah Layanan Berbasis Lokasi, atau yang lebih akrab disebut sebagai Location Based Service (LBS), mengacu pada teknologi yang digunakan untuk melacak posisi perangkat kita. LBS adalah layanan informasi yang bisa diakses melalui perangkat mobile dengan bantuan jaringan seluler, yang dapat menggunakan lokasi dari perangkat tersebut. Berikut dua unsur utama dalam LBS yaitu:

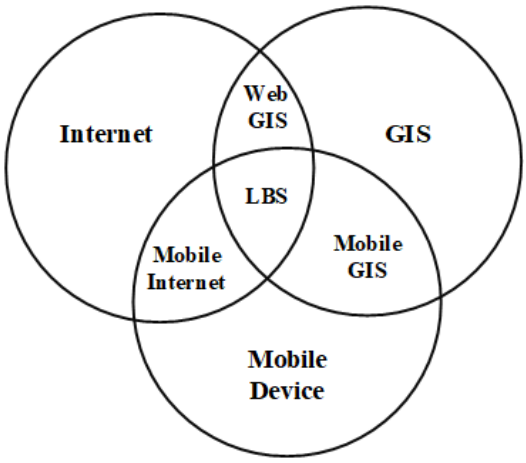
1. Location Manager (API Maps)

   Menyediakan alat atau sumber untuk Layanan Berbasis Lokasi (LBS), Application Programming Interface (API) Maps memberikan kemudahan dalam menampilkan dan mengelola peta, serta fitur-fitur tambahan seperti mode tampilan satelit, jalanan, dan kombinasinya. Komponen ini terdapat dalam paket com.google.android.maps.

2. Location Providers (API Location)

   Location Provider adalah teknologi yang disediakan oleh perangkat untuk mencari lokasi. API Location terhubung dengan data GPS (*Global Positioning System*) dan data lokasi real-time, yang terletak dalam paket android.location. Dengan menggunakan Location Manager, kita bisa menetapkan lokasi saat ini, melacak pergerakan, serta mendeteksi kedekatan dengan lokasi tertentu berdasarkan perubahan lokasi.

   Berikut salah satu Teknologi Location Based Service dapat dilihat pada gambar 2.3:



Gambar 0.3 Teknologi Location Based Service

Location Based Service (LBS) merupakan layanan yang beroperasi di persilangan tiga teknologi utama: Geographic Information System (GIS), Internet Service, dan Mobile Devices. LBS berkaitan dengan cara menemukan posisi dari perangkat yang digunakan, juga dikenal sebagai metode positioning. Ada tiga jenis sistem positioning yang umum digunakan dalam LBS, yaitu:

1. Metode manual adalah pendekatan konvensional yang sering digunakan sebelumnya, seperti melalui buku kuning, layanan operator telepon, dan sebagainya. Cara-cara ini kini dianggap kuno dan kurang efisien. Kemunculan internet membawa perubahan signifikan dalam cara kita melakukan pencarian informasi, menjadi lebih luas dan mudah diakses. Dukungan dari teknologi komunikasi seluler juga memperluas mobilitas dan kenyamanan akses. Gabungan kedua teknologi ini telah mengarah pada peningkatan signifikan dalam efisiensi pencarian, menggantikan metode manual yang kurang efisien.
2. GPS (*Global Positioning System*) adalah sistem navigasi global berbasis radio yang mengandalkan 24 satelit dan stasiun bumi. Sistem ini membagi bumi menjadi segi empat dengan alamat unik untuk setiap lokasi, memungkinkan identifikasi yang akurat terhadap lokasi tersebut.
3. Melalui *Base Transceiver Station* (BTS) atau Stasiun Pemancar Seluler, teknologi ini berbasis pada jaringan telekomunikasi seluler yang memungkinkan penentuan lokasi di dalam ruangan atau indoor. Dengan menggunakan BTS, posisi handphone dapat diidentifikasi berdasarkan sinyal yang diterima dari satu atau lebih cell tower terdekat, serta menggunakan prinsip triangulasi. Meskipun

demikian, akurasi penentuan lokasi dengan Cellular Based Station cenderung lebih rendah daripada GPS.

Dalam konteks Layanan Berbasis Lokasi, ada lima elemen kunci yang harus dipertimbangkan, yakni:

1. *Mobile Device* merupakan perangkat yang digunakan untuk mengakses informasi yang dibutuhkan oleh pengguna. Jenis perangkat yang termasuk dalam kategori ini meliputi PDA, ponsel pintar, laptop, dan perangkat lainnya yang dilengkapi dengan fitur navigasi. Dengan Mobile Devices, pengguna dapat meminta informasi dalam berbagai bentuk seperti suara, gambar, dan teks.
2. *Communication Network* adalah sistem yang mengirimkan data dari pengguna dan menerima permintaan layanan. Jaringan ini memfasilitasi aliran data antara terminal mobile dan penyedia layanan, serta mengirimkan balik informasi yang diminta kepada pengguna. Jaringan Komunikasi dapat berupa jaringan seluler seperti GSM atau CDMA, jaringan Wireless Local Area Network (WLAN), atau jaringan Wireless Wide Area Network (WWAN).
3. *Positioning Component* diperlukan dalam pengolahan layanan karena posisi pengguna harus diketahui terlebih dahulu. Posisi pengguna dapat ditentukan melalui jaringan komunikasi atau menggunakan Global Positioning System (GPS). Selain GPS, posisi juga dapat diambil melalui Cell Tower atau kombinasi GPS dan Cell Tower (aGPS).
4. *Service and Application Provider* adalah pihak yang menyediakan layanan bagi pengguna seluler dan bertanggung jawab atas pemrosesan layanan tersebut. Provider ini melakukan berbagai proses komputasi, seperti menemukan rute perjalanan, mencari informasi lokasi terdekat, dan mengakses database eksternal seperti yellow pages atau Google API untuk menghasilkan informasi yang diminta oleh pengguna.
5. *Data and Content Provider* merupakan penyedia layanan informasi yang dapat diminta oleh pengguna. Mereka tidak selalu menyimpan semua data yang diolah, karena data tersebut dapat berasal dari pengembang atau pihak ketiga yang memiliki otoritas menyimpannya. Contohnya, data geografis dapat berasal dari badan pemerintah, sementara data bisnis dapat berasal dari Yellow Pages atau

penyedia data lainnya. Sebagai penyedia data, mereka memfasilitasi pengguna dalam mengakses informasi yang dibutuhkan.

1.10 Maps SDK for Android

Google Maps SDK for Android adalah sebuah alat pengembangan yang memungkinkan pengembang untuk menintegrasikan peta Google ke aplikasi Android mereka. Dengan Google Maps SDK for Android, pengembang dapat menampilkan peta interaktif, menambahkan marker, melakukan zoom dan pemindahan peta, serta mengimplementasikan fitur-fitur lainnya seperti rute, lokasi, dan pencarian. Selain itu, Google Maps SDK for Android juga menyediakan fitur geocoding dan reverse geocoding, yang memungkinkan pengembang untuk mengubah alamat ke koordinat lokasi dan sebaliknya[21].

1.11 Firebase

Firebase adalah sebuah kerangka kerja yang sangat berguna dalam pembangunan aplikasi seluler dan web, terutama bagi bisnis yang memerlukan basis data *real-time*. Diciptakan oleh Andrew Lee dan James Tamplin pada tahun 2011 dan resmi diluncurkan pada bulan April 2012, Firebase awalnya dirancang sebagai basis data *real-time* dengan antarmuka pemrograman aplikasi (API) yang memungkinkan pengguna untuk menyimpan dan menyinkronkan data di antara pengguna lainnya. Setelah diakuisisi oleh Google pada tahun 2014, Firebase berkembang menjadi sebuah layanan yang menawarkan berbagai fitur dan alat pengembangan bagi para pengembang dan pengusaha. Melalui basis data *real-time*, Firebase memastikan bahwa pembaruan data yang dilakukan oleh satu pengguna dapat segera diteruskan kepada pengguna lainnya. Firebase juga menyediakan platform yang sederhana dan terintegrasi untuk berbagai aplikasi, serta menawarkan berbagai fitur tambahan dari Google. Dengan mengurangi beban kerja server-side dalam pengembangan aplikasi, Firebase menjadi sebuah alat yang penting bagi para pengembang, dengan berbagai elemen yang mempermudah proses pengembangan aplikasi dan memastikan keseimbangan antara pengembang dan pengguna akhir dengan mengurangi keterlambatan kerja[7].

Berikut beberapa komponen atau layanan yang ditawarkan oleh Firebase adalah sebagai berikut:

1.11.1 Authentication

Firebase Authentication menawarkan layanan backend, SDK yang simpel untuk digunakan, dan perpustakaan UI siap pakai untuk memverifikasi pengguna di dalam aplikasi Anda. Layanan ini mendukung autentikasi dengan berbagai cara, termasuk sandi, email atau nama pengguna, nomor telepon, dan metode lainnya. Pengguna dapat login ke aplikasi Firebase menggunakan FirebaseUI sebagai opsi autentikasi komprehensif atau dengan menggunakan SDK untuk mengintegrasikan teknik login secara manual[22]. Adapun contoh source code membuat akun baru dapat dilihat pada Gambar 2.4 berikut.

```
mAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's
information
                    Log.d(TAG, "createUserWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "createUserWithEmail:failure",
task.getException());
                    Toast.makeText(EmailPasswordActivity.this, "Authentication
failed.",
                            Toast.LENGTH_SHORT).show();
                    updateUI(null);
                }
            }
        });
```

Gambar 0.4 Contoh Source Code Membuat Akun Baru

Sumber: https://firebase.google.com/docs/auth/android

Adapun contoh source code login pengguna yang ada dapat dilihat pada Gambar 2.5 berikut.

```
mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's
information
                    Log.d(TAG, "signInWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "signInWithEmail:failure", task.getException());
                    Toast.makeText(EmailPasswordActivity.this, "Authentication
failed.",
                            Toast.LENGTH_SHORT).show();
                    updateUI(null);
                }
```

```
17.             }
18.         });
```

Gambar 0.5 Contoh Source Code Login Pengguna

Sumber: https://firebase.google.com/docs/auth/android

1.11.2 Realtime Database

Firebase Realtime Database merupakan sebuah database yang dihosting di cloud. Data-data disimpan dalam format JSON dan disinkronkan secara terus menerus ke setiap klien yang terhubung. Saat mengembangkan aplikasi lintas platform menggunakan SDK iOS, Android, dan JavaScript, semua klien akan menggunakan satu instance Realtime Database yang sama dan mendapatkan pembaruan data secara otomatis. Fitur ini memungkinkan pengembang untuk menghindari langkah pengembangan database, karena Firebase menangani sebagian besar backend aplikasi. Firebase juga menyediakan bahasa aturan yang fleksibel dan berbasis ekspresi untuk mendefinisikan bagaimana data harus diorganisir dan kapan informasi dapat dibaca atau ditulis[23]. Adapun contoh source code untuk menulis data dapat dilihat pada Gambar 2.6 berikut.

```
1. // Write a message to the database
2. FirebaseDatabase database = FirebaseDatabase.getInstance();
3. DatabaseReference myRef = database.getReference("message");
4.
5. myRef.setValue("Hello, World!");
```

Gambar 0.6 Contoh Source Code Menulis Data

Sumber: https://firebase.google.com/docs/database/android

Adapun contoh source code membaca data dapat dilihat pada Gambar 2.7 berikut.

```
 1. // Read from the database
 2. myRef.addValueEventListener(new ValueEventListener() {
 3.     @Override
 4.     public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
 5.         // This method is called once with the initial value and again
 6.         // whenever data at this location is updated.
 7.         String value = dataSnapshot.getValue(String.class);
 8.         Log.d(TAG, "Value is: " + value);
 9.     }
10.
11.     @Override
12.     public void onCancelled(@NonNull DatabaseError error) {
13.         // Failed to read value
14.         Log.w(TAG, "Failed to read value.", error.toException());
15.     }
16. });
```

Gambar 0.7 Contoh Source Code Membaca Data

Sumber: https://firebase.google.com/docs/database/android

1.11.3 Cloud Messaging

Cloud Messaging adalah solusi lintas platform yang memungkinkan pengembang untuk mengirimkan pesan dengan biaya nol secara dapat diandalkan. Pengembang dapat mengirimkan pesan notifikasi untuk mendorong keterlibatan dan retensi pengguna[7]. API harus memiliki kemampuan untuk mengakses database sistem informasi yang sedang berjalan dan mampu mengirimkan notifikasi kepada orang tua yang telah terdaftar di aplikasi. Untuk mengatasi tantangan tersebut, digunakan teknologi cloud messaging yang dapat memberikan informasi secara langsung ke pengguna tanpa biaya tambahan[24]. Salah satu platform yang menyediakan layanan cloud messaging adalah Firebase Cloud Messaging (FCM), sebelumnya dikenal sebagai Google Cloud Messaging (GCM). Melalui FCM, aplikasi pengguna dapat diberitahu tentang adanya email baru atau data lain yang perlu disinkronkan. Notifikasi juga dapat digunakan untuk mendorong interaksi aktif dari pengguna dan mempertahankan keterlibatan mereka. Adapun contoh source code FirebaseMessagingService disimpan didalam Manifest untuk menangani pesan dan menerima notifikasi aplikasi saat berjalan di latar belakang dapat dilihat pada Gambar 2.8 berikut.

```
1. <service
2.     android:name=".java.MyFirebaseMessagingService"
3.     android:exported="false">
4.     <intent-filter>
5.         <action android:name="com.google.firebase.MESSAGING_EVENT" />
6.     </intent-filter>
7. </service>
```

Gambar 0.8 Contoh Source Code FirebaseMessagingService

Sumber: https://firebase.google.com/docs/cloud-messaging/android

Adapun contoh source code untuk mengambil toke FCM agar dapat dikirim pesan dari FCM dapat dilihat pada Gambar 2.9 berikut.

```
 1. FirebaseMessaging.getInstance().getToken()
 2.     .addOnCompleteListener(new OnCompleteListener<String>() {
 3.         @Override
 4.         public void onComplete(@NonNull Task<String> task) {
 5.           if (!task.isSuccessful()) {
 6.             Log.w(TAG, "Fetching FCM registration token failed",
task.getException());
 7.             return;
 8.           }
 9.
10.           // Get new FCM registration token
11.           String token = task.getResult();
12.
```

```
13.           // Log and toast
14.           String msg = getString(R.string.msg_token_fmt, token);
15.           Log.d(TAG, msg);
16.           Toast.makeText(MainActivity.this, msg, Toast.LENGTH_SHORT).show();
17.         }
18.     });
```

Gambar 0.9 Contoh Source Code Mengambil Token FCM

Sumber: https://firebase.google.com/docs/cloud-messaging/android

Adapun contoh source code menerima pesan dari FCM agar aplikasi dapat menerima pesan dari FCM dapat dilihat pada Gambar 2.10 berikut.

```
 1. @Override
 2. public void onMessageReceived(RemoteMessage remoteMessage) {
 3.     // TODO(developer): Handle FCM messages here.
 4.     // Not getting messages here? See why this may be: https://goo.gl/39bRNJ
 5.     Log.d(TAG, "From: " + remoteMessage.getFrom());
 6.
 7.     // Check if message contains a data payload.
 8.     if (remoteMessage.getData().size() > 0) {
 9.         Log.d(TAG, "Message data payload: " + remoteMessage.getData());
10.
11.         if (/* Check if data needs to be processed by long running job */ true)
{
12.             // For long-running tasks (10 seconds or more) use WorkManager.
13.             scheduleJob();
14.         } else {
15.             // Handle message within 10 seconds
16.             handleNow();
17.         }
18.
19.     }
20.
21.     // Check if message contains a notification payload.
22.     if (remoteMessage.getNotification() != null) {
23.         Log.d(TAG, "Message Notification Body: " +
remoteMessage.getNotification().getBody());
24.     }
25.
26.     // Also if you intend on generating your own notifications as a result of a
received FCM
27.     // message, here is where that should be initiated. See sendNotification
method below.
28. }
```

Gambar 0.10 Contoh Kode Menerima Pesan Dari FCM

Sumber: https://firebase.google.com/docs/cloud-messaging/android

1.11.4 Cloud Firestore

Cloud Firestore adalah sebuah layanan database yang tersedia di Google Cloud Platform. Ini memudahkan pengembang, untuk mengintegrasikan dan memanfaatkan kekuatan penyimpanan dan manajemen data yang ditawarkan oleh Google Cloud Platform melalui Firebase SDK. Cloud Firestore bukan layanan langsung dari Firebase, melainkan bagian dari Google Cloud Platform. Firebase menyediakan SDK agar kita bisa menggunakan Cloud Firestore dengan mudah[25].

Adapun contoh source code untuk menambahkan data dapat dilihat pada Gambar 2.11 berikut.

```
// Create a new user with a first and last name
Map<String, Object> user = new HashMap<>();
user.put("first", "Ada");
user.put("last", "Lovelace");
user.put("born", 1815);

// Add a new document with a generated ID
db.collection("users")
        .add(user)
        .addOnSuccessListener(new OnSuccessListener<DocumentReference>() {
            @Override
            public void onSuccess(DocumentReference documentReference) {
                Log.d(TAG, "DocumentSnapshot added with ID: " +
documentReference.getId());
            }
        })
        .addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Log.w(TAG, "Error adding document", e);
            }
```

Gambar 0.11 Contoh Source Code Menambahkan Data

https://firebase.google.com/docs/firestore/

Adapun contoh source code untuk membaca data dapat dilihat pada Gambar 2.12 berikut.

```
db.collection("users")
        .get()
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            @Override             public void onComplete(@NonNull
Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    for (QueryDocumentSnapshot document : task.getResult()) {
                        Log.d(TAG, document.getId() + " => " +
document.getData());
                    }
                } else {
                    Log.w(TAG, "Error getting documents.", task.getException());
                }
            }
        });
```

Gambar 0.12 Contoh Source Code Membaca Data

https://firebase.google.com/docs/firestore/

1.11.5 Cloud Storage

Cloud Storage for Firebase adalah layanan penyimpanan file yang andal, mudah digunakan, dan terjangkau dari Google. Layanan ini menyediakan tempat untuk menyimpan file dalam jumlah besar, yang disebut "bucket". Firebase

menyediakan SDK yang memungkinkan developer untuk mengupload dan mendownload file dengan aman, terlepas dari kualitas jaringan, langsung dari aplikasi mereka[25]. Adapun contoh source code membuat referensi untuk penanda atau tautan ke file dalam cloud yang memungkinkan Anda untuk mengunggah, mengunduh, menghapus file, atau memperoleh dan memperbarui metadatanya dapat dilihat pada Gambar 2.13 berikut.

```
1. // Create a child reference
2. // imagesRef now points to "images"
3. StorageReference imagesRef = storageRef.child("images");
4.
5. // Child references can also take paths
6. // spaceRef now points to "images/space.jpg
7. // imagesRef still points to "images"
8. StorageReference spaceRef = storageRef.child("images/space.jpg");
```

Gambar 0.13 Contoh Source Code Membuat Referensi

https://firebase.google.com/docs/storage/android/

1.12 API (*Application Programming Interface*)

API adalah alat yang menyederhanakan komunikasi antara aplikasi dan server, membantu dalam implementasi dan pengembangan perangkat lunak. API terdiri dari perintah, fungsi, dan protokol yang memfasilitasi interaksi programmer dengan sistem operasi, memungkinkan penggunaan fungsi standar untuk berkomunikasi dengan sistem tersebut. API juga dapat diartikan sebagai sekumpulan protokol yang mengatur interaksi antara aplikasi untuk pertukaran informasi dengan bahasa yang sama[26].

1.13 JSON (*JavaScript Object Notation*)

JSON merupakan pertukaran data komputer yang sederhana dan mudah dibaca manusia, berbasis teks. Format ini digunakan untuk merepresentasikan struktur data sederhana yang dikenal sebagai objek. JSON sering digunakan dalam mentransmisikan data terstruktur melalui jaringan, melalui proses yang disebut encoding. Aplikasi utamanya adalah dalam pengembangan aplikasi web AJAX sebagai alternatif terhadap format XML yang lebih tradisional.

1.14 XML

Extensible Markup Language, atau dikenal sebagai XML, adalah sebuah format file dan bahasa markup yang berguna untuk menyimpan, mengirim, dan merekonstruksi data dengan beragam konten. XML memberikan aturan baku dalam mengkodekan dokumen sehingga bisa dimengerti oleh mesin dan manusia. Ini

menggunakan format teks yang fleksibel dan mudah dimengerti, berasal dari pengembangan SGML (ISO 8879). XML digunakan secara luas untuk menyusun data-data penting seperti catatan basis data, transaksi, dan berbagai jenis informasi lainnya.

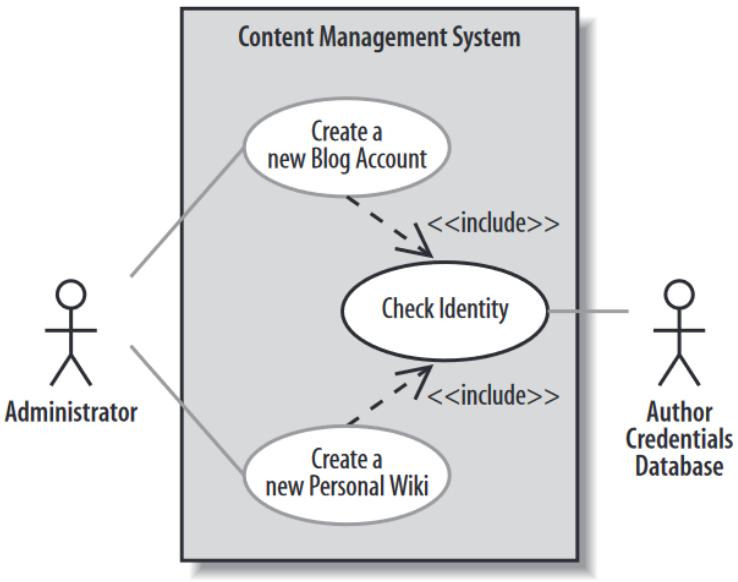
1.15 UML

Unified Modeling Language (UML) adalah bahasa pemodelan serbaguna yang memberikan cara konsisten untuk menggambarkan desain sistem secara visual. UML menggunakan notasi visual untuk menggambarkan bagaimana sistem berinteraksi dan terstruktur, bukan sebagai bahasa pemrograman. Hal ini membantu insinyur perangkat lunak, pebisnis, dan arsitek sistem dalam pemodelan, desain, dan analisis sistem. UML menyediakan berbagai jenis diagram, termasuk diagram perilaku, diagram interaksi, dan diagram struktur, yang memiliki notasi standar untuk mempermudah pemahaman. Dalam perancangan sistem ini, penulis menggunakan Unified Modeling Language (UML) yang mencakup Use Case Diagram, Activity Diagram, Class Diagram, dan Desain Input-Output. Hal ini dilakukan untuk menciptakan sistem informasi yang terstruktur dan terorganisir dengan baik[27].

1.15.1 Use Case Diagram

Use case diagram menggambarkan apa yang diharapkan dari sebuah sistem dari sudut pandang fungsional. Fokusnya adalah pada aktivitas atau tugas yang dilakukan oleh sistem, bukan bagaimana cara melakukannya. Setiap use case mewakili interaksi antara pengguna atau aktor dengan sistem untuk mencapai tujuan tertentu, seperti login atau membuat daftar belanja. Aktor bisa berupa manusia atau entitas mesin yang berinteraksi dengan sistem untuk menyelesaikan tugas-tugas khusus[28].

Berikut salah satu sampel Use Case Diagram dapat dilihap pada gambar 2.14:



Gambar 0.14 Use Case Diagram

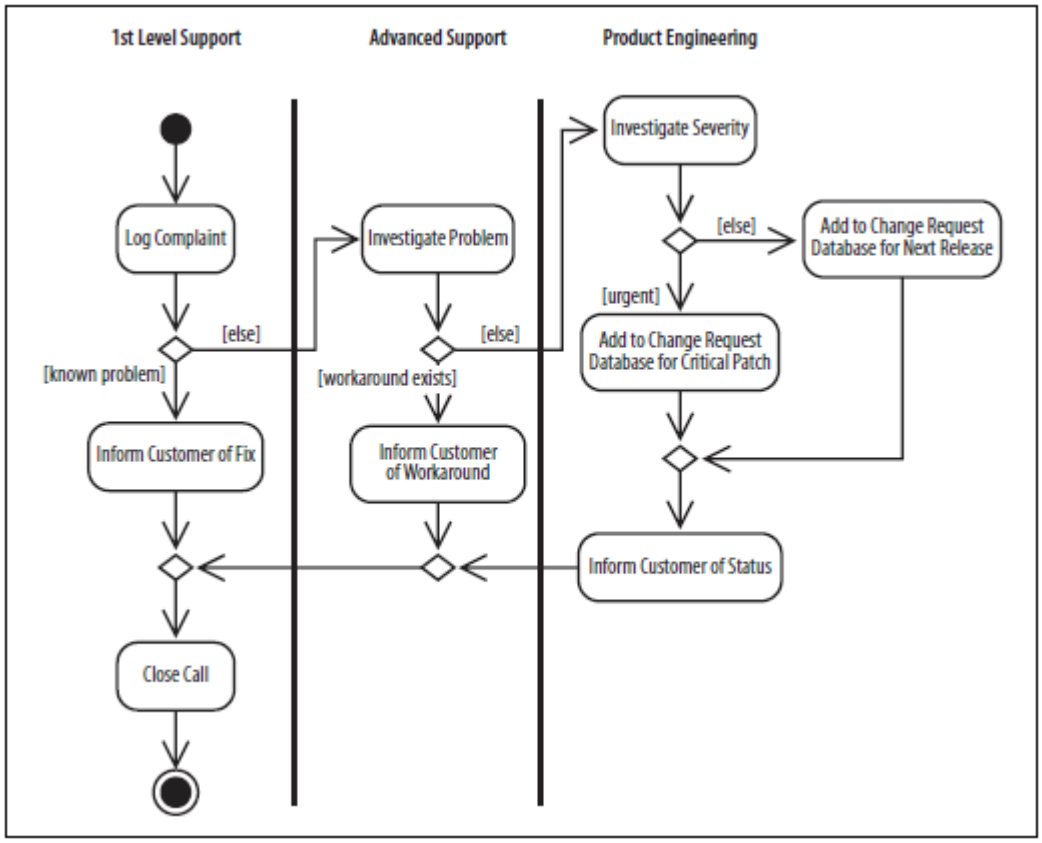
Dalam gambar di atas, terdapat dua elemen penting yaitu aktor yang merupakan pengguna sistem, dan Use Case yang mencakup fungsi-fungsi dalam sistem. Use Case memodelkan kebutuhan sistem dengan jelas dan mengidentifikasi kebutuhan pengguna. Karena Use Case mewakili kebutuhan fungsional sistem secara rinci, penting untuk memberikan perhatian khusus pada tahap awal sebelum melanjutkan ke tahap pemodelan lainnya. Setelah merancang kebutuhan fungsional sistem, langkah berikutnya adalah menentukan skenario-skenario yang terdapat dalam Use Case, yang menjelaskan kondisi awal hingga akhir yang harus dipenuhi oleh suatu Use Case. Dengan demikian, Use Case Diagram berperan penting dalam menentukan kemampuan sistem[29].

1.15.2 Activity Diagram

Setelah menetapkan kebutuhan fungsional sistem, langkah berikutnya adalah memahami urutan aktivitas yang terjadi dalam suatu Use Case dari awal hingga mencapai tujuan akhirnya. Untuk mengatasi tantangan ini, digunakan Activity Diagram sebagai solusi. Activity Diagram membantu menggambarkan secara visual urutan langkah-langkah atau aktivitas yang dilakukan dalam suatu proses atau Use Case, sehingga memudahkan pemahaman tentang bagaimana sistem bekerja dari awal hingga selesai. Diagram aktivitas mengilustrasikan serangkaian aktivitas dalam sistem, termasuk bagaimana aktivitas dimulai,

keputusan yang dapat muncul, dan bagaimana mereka selesai. Diagram ini juga dapat menunjukkan aktivitas paralel yang mungkin terjadi dalam beberapa situasi eksekusi[28].

Berikut sampel Activity Diagram dapat dilihat pada gambar 2.15:



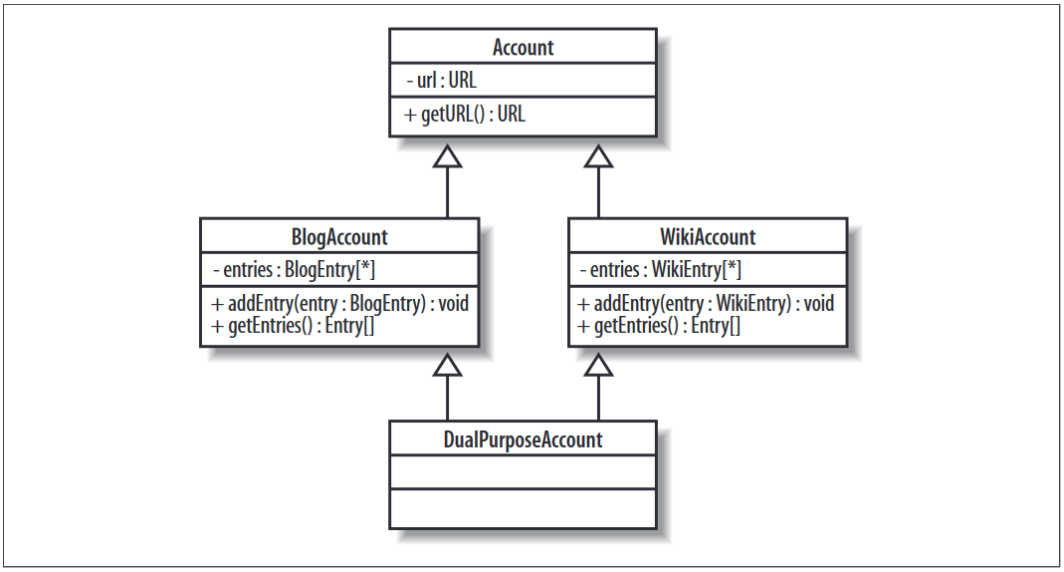
Gambar 0.15 Activity Diagram

Diagram aktivitas berfungsi untuk menggambarkan urutan aktivitas dari suatu Use Case dengan detail, mulai dari awal hingga akhir. Contohnya, kita dapat menggunakan diagram aktivitas untuk menunjukkan langkah-langkah dalam membuat proses registr. Dari penjelasan ini, kita dapat mengerti bahwa diagram aktivitas sangat berguna dan sesuai untuk memodelkan aturan bisnis di dalam suatu sistem dengan praktis dan detail[29].

1.15.3 Class Diagram

Class Diagram merupakan spesifikasi yang saat dijadikan objek akan menciptakan instansiasi dan menjadi dasar dari desain berorientasi objek. Class memperlihatkan atribut sistem dan memberikan fungsi untuk mengelola atribut tersebut. Dan juga ada sebuah layanan yang bisa merubah suatu keadaan seperti metode[28]. Dalam pemrograman berbasis objek (OOP), Class merupakan elemen

kunci. Setiap sistem OOP memecahkan masalahnya dengan susunan kelas yang saling terhubung. Setiap kelas memiliki atribut dan metode yang berkontribusi untuk mencapai tujuannya. Sistem OOP yang baik akan memiliki banyak kelas yang terorganisir dengan baik. Tanpa organisasi yang tepat, pengelolaan dan pengembangan sistem akan sulit. UML menyediakan solusi dengan Class Diagram untuk menggambarkan peran dan relasi antar kelas. Berikut salah satu sampel class diagram dapat dilihat pada Gambar 2.16 berikut.



Gambar 0.16 Sampel Class Diagram

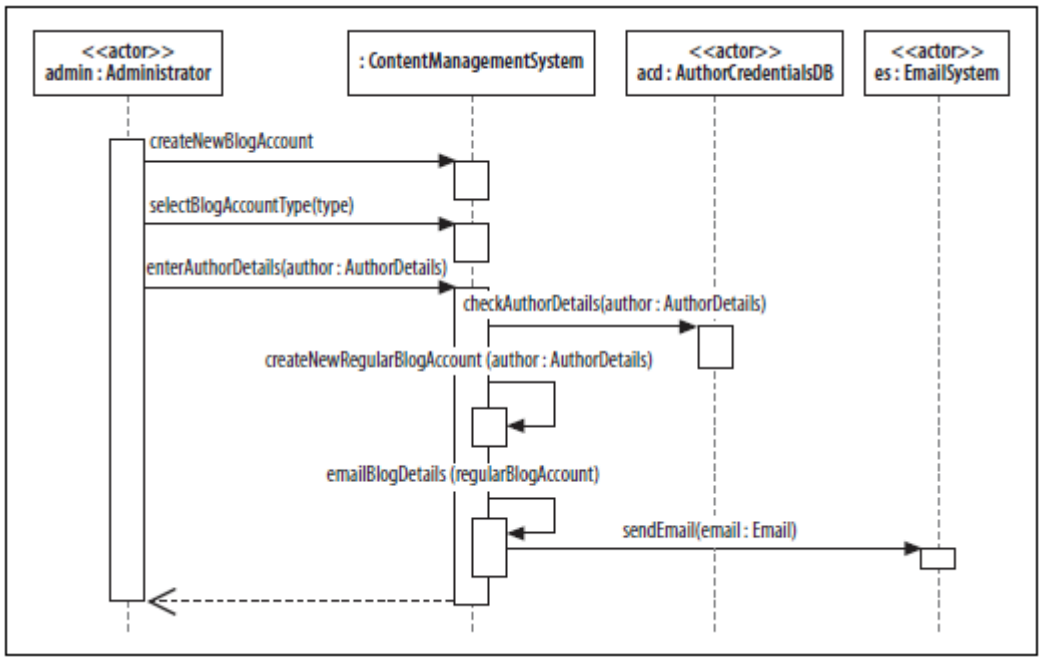
Class Diagram merupakan alat yang sangat berguna untuk menggambarkan hubungan antar kelas dalam suatu sistem. Diagram ini dapat menggambarkan berbagai jenis hubungan seperti *Dependency*, *Aggregation*, dan *Association*[29].

1.15.4 Sequence Diagram

Sequence diagram merupakan alat yang digunakan untuk menunjukkan interaksi antara objek dalam sistem, termasuk pengguna, tampilan, dan sebagainya, melalui pesan yang ditunjukkan dalam urutan waktu. Diagram ini memiliki dimensi vertikal yang mewakili waktu dan dimensi horizontal yang menunjukkan objek yang terlibat dalam interaksi tersebut. Sequence diagram digunakan untuk menggambarkan rangkaian langkah atau skenario yang terjadi sebagai respons dari suatu kejadian, menggambarkan proses internal yang terjadi dan hasil output yang dihasilkan sebagai akibat dari interaksi tersebut[28]. Sequence Diagram juga adalah

gambaran visual dari bagaimana kelas-kelas berinteraksi di dalam sistem, disusun dalam urutan waktu dari atas ke bawah. Dimulai dari awal interaksi hingga akhir interaksi, diagram ini menggambarkan urutan langkah-langkah yang terjadi antara kelas-kelas tersebut.

Dapat dilihat sampel Sequence Diagram pada gambar 2.17:



Gambar 0.17 Sequence Diagram

Sequence Diagram adalah representasi grafis dari urutan aktivitas dalam sistem, memperlihatkan komunikasi antar objek hingga mencapai tujuan akhir. Dari Gambar 2.8 diatas diagram ini menggunakan garis waktu untuk setiap objek, dengan setiap interaksi ditunjukkan melalui anak panah yang disebut message. Interaksi ini terus berlangsung hingga tujuan akhir tercapai[29].