BAB 2
LANDASAN TEORI

2.1 Kalori

Kalori merupakan unit energi yang digunakan untuk mengukur energi yang diperoleh dari makanan. Dalam konteks kebutuhan kalori harian, kalori mencerminkan kemampuan makanan untuk memberikan energi, baik secara biokimia maupun fisik. Asupan kalori juga berpengaruh pada pertumbuhan atau penurunan berat badan seseorang, di mana asupan kalori yang berlebihan dapat menyebabkan obesitas dan menghambat aktivitas fisik.[17]

Resting Metabolic Rate (RMR) merupakan kebutuhan energi atau kalori minimal yang diperlukan oleh tubuh untuk menjalankan proses dasar, seperti peredaran darah, pernapasan, dan metabolisme sel, dan mempertahankan suhu tubuh.[10]

2.2 Joging

Joging merupakan aktivitas olahraga berlari pelan untuk aktivitas kesehatan dan berat badan ideal. Hal tersebut dibuktikan bahwa joging dapat menguranngi berat badan [12]. Joging juga dapat menurunkan gejala diabetes serta menguatkan otot jantung [18]. Oleh karena itu, joging dipilih sebagai olahraga yang efektif baik dalam menyehatkan tubuh dan efektif dalam penurunan berat badan.

Adapun kalori kalori yang dibakar untuk mengurangi berat badan 0,5 kg dalam waktu satu minggu yaitu dengan cara melakukan joging dengan target pembakaran kalori 500 *kcal* per hari sehingga penurunan berat badan dapat berlangsung cepat. Tetapi, hal ini dapat dilakukan jika penguranran kalori terhadap makanan tidak banyak dipangkas.[6]

Adapun aturan untuk diet atau pengurangan kalori maksimum adalah 800 *kcal* per hari. Jadi diet akan dilakukan dengan cara menekan 275 *kcal* dari diet karbohidrat serta melakukan joging dengan pembakara kalori 500 *kcal* untuk mengurangi 1 *pound* sampai 1,5 *pounds* per minggu.[6], [19].

2.3 *Diet*

Diet merupakan cara mengkombinasi makanan yang dikonsumsi serta mengatur konsumsi makanan setiap hari dengan tujuan berat badan yang ideal [20]. Salah

satu diet yang efektif untuk menurunkan berat badan adalah diet rendah karbohidrat [10], [19]. Diet karbohidrat dapat dilakukan dengan cara mengurangi konsumsi karbohidrat. Diet rendah karbohidrat dan tinggi protein dapat menurunkan berat badan yang efektif dalam jangka waktu 6 bulan sampai dengan 1 tahun. Adapun batas dari diet yang dilakukan yaitu sebesar 800 *kcal* [19]. Diet dengan pengurangan kalori maksimum sebesar 800 *kcal* dapat dibantu dengan joging sehingga proses diet dapat berlangsung lebih cepat [12]. Adapun untuk peningkatan berat badan bagi pengguna yang memiliki berat badan di bawah ideal dapat melakukan konsumsi makanan yang tinggi protein agar berat badan bertambah sebab pertumbuhan otot yang meningkat [16].

2.4 *Body Mass Index (BMI)*

BMI merupakan indikator pengukuran yang digunakan untuk menentukan kategori berat badan ideal atau tidak. Indikator tersebut hanya bisa digunakan untuk orang yang berusia 18 tahun ke atas karena riset yang dilakukan hanya fokus kepada pria dan Wanita dewasa [21]. Indeks *BMI* yang digunakan merupakan indeks bmi yang diterbitkan oleh *WHO [5].* Indeks *BMI* dikategorikan menurut *WHO* adalah sebagai berikut:

1. Kurus Sekali: *BMI* dengan nilai kurang dari kurang dari 16,0, maka berat badan dikategorikan Kurus Sekali.
2. Kurus: *BMI* dengan nilai mulai dari 16,0 sampai dengan 16,99, maka berat badan dikategorikan Sedikit Kurus.
3. Sedikit Kurus: *BMI* dengan nilai mulai dari 17,0 sampai dengan 18,49 maka berat badan dikategorikan Sedikit Kurus.
4. Ideal: *BMI* dengan nilai mulai dari 18,5 sampai dengan 24,49, maka berat badan dikategorikan normal.
5. Berlebih: *BMI* dengan nilai mulai dari 25 sampai dengan 29,99 maka berat badan dikategorikan *overweight* atau berat badan berlebih.
6. Obesitas Kelas 1: *BMI* dengan nilai mulai dari 30 sampai dengan 34,99 maka berat badan dikategorikan sebagai Obesitas Kelas 1.
7. Obesitas Kelas 2: *BMI* dengan nilai lebih dari 35 sampai dengan 39,99 maka berat badan dikategorikan sebagai Obesitas Kelas 2.

8. Obesitas Kelas 3: *BMI* dengan nilai mulai dari 40 ke atas maka berat badan dikategorikan sebagai Obesitas Kelas 3.

Adapun persamaan indeks *BMI* adalah sebagai berikut:

$$BMI = \frac{W}{T^2} \quad \textbf{(1)}$$

Keterangan:

$W = Berat\ badan\ (kg)$

$T = Tinggi\ badan\ (m)$

Rumus di atas digunakan untuk mengkategorikan berat badan apakah berat badan tersebut ideal atau berlebih.[5]

2.5 *Resting Metabolic Rate (RMR)*

RMR merupakan metode persamaan untuk menghitung seberapa banyak kalori yang diperlukan seseorang selama satu hari [22]. *RMR* dapat dihitung berdasarkan jenis kelamin dan berat badan serta usia. Perhitungan *RMR* untuk laki-laki dan perempuan berbeda, di mana kebutuhan kalori perempuan lebih rendah dibandingkan dengan kebutuhan kalori laki-laki. Rumus *RMR* sebagai berikut:

1. Laki-laki

Berikut rumus *RMR* untuk laki-laki:

$$RMR = 260 + (9{,}65\ x\ W) + (573\ X\ T) - (5.08\ x\ A) \quad \textbf{(2)}$$

Keterangan:

$W = Berat\ badan\ (kg)$

$T = Tinggi\ badan\ (m)$

$A = Usia\ (tahun)$

2. Perempuan

Berikut rumus *RMR* untuk Perempuan:

$$RMR = 43 + (7{,}38\ x\ W) + (607\ x\ T) - (2{,}31\ x\ A) \quad \textbf{(3)}$$

Keterangan:

$W = Berat\ badan\ (kg)$

$T = Tinggi\ badan\ (cm)$

$A = Usia\ (tahun)$

2.6 *Metabolic Equivalent of Task (MET)*

MET merupakan suatu nilai dari pengukuran dari seberapa banyak oksigen yang dibakar saat tubuh melakukan aktivitas kecil [13]. *MET* juga bisa dihubungkan dengan seberapa banyak kalori yang dibakar selama melakukan aktivitas sehingga kalori yang dibuang dapat diketahui. Nilai *MET* yang digunakan adalah 2,8 untuk olahraga joging. Nilai 2,8 tersebut digunakan untuk menghitung kalori yang terbuang setelah melakukan aktivitas jogging [23]. Persamaan *MET* khusus untuk Joging sebagai berikut:

$$EC = MET\ x\ 7{,}7\ x\ \left(\frac{(\ W\ x\ 2{,}2\ )}{200}\right)\ x\ \frac{Duration}{60} \qquad \textbf{(4)}$$

Keterangan:

EC = *Exercise Calories* yang menghasilkan nilai kalori terbakar di saat joging.

MET = 2,8

W = Berat badan (kg)

Duration = durasi olahraga (detik)

2.7 *Android*

Android merupakan sistem operasi berbasis Linux yang dirancang untuk perangkat ponsel cerdas layar sentuh dan komputer tablet. *Android* dikembangkan oleh *Android*, *Inc*. dengan tujuan untuk membuat sistem operasi untuk perangkat mobile. Di zaman ini, *Android* sudah berkembang menjadi sistem operasi yang menyediakan berbagai macam fitur dan terkoneksi dengan berbagai fasilitas dari *Google*, yang merupakan induk perusahaan dari *Android*, *Inc*.

Versi *Android* merupakan satu *tag* yang terdapat pada A*ndroid* yang dirilis dari tahun 2007 awal terbit sampai dengan sekarang. Biasanya versi *android* ini diberi nama makanan seperti *Donut*, *Gingerbread*, dan sebagainya. Versi *Android* yang terbaru adalah *Tiramisu*. Oleh karena itu perlu dipertimbangkan seberapa banyak perangkat *Android* yang dapat menjalankan *software* yang akan dibuat sehingga diputuskan untuk menggunakan *api* level 22 sehingga masih ada sekitar 98.8% perangkat yang masih menggunakan *api* level 22 ke atas.

2.7.1 *Kotlin*

Bahasa pemrograman yang digunakan adalah *Kotlin* dimana *Kotlin* memang dikembangkan oleh *Google* dan *Jetbrains* untuk mempersingkat proses

pembangunan aplikasi serta membuat *developer* dapat mengakses kode program java ataupun bahasa pemrograman lainnya serta mempermudah *developer* untuk membuat kode program yang aman [24]. *Kotlin* merupakan bahasa pemrograman yang dikembangkan oleh komunitas terbuka *(Open source) [25].*

2.8 Application Programming Interface

API (Application Programming Interface) merupakan suatu perangkat lunak yang bertujuan untuk membuka komunikasi dengan perangkat lunak lain agar perangkat lunak tersebut mendapatkan informasi dari suatu *service* atau layanan. Didalam sebuah *API* juga terdapat dokumentasi yang membantu pengembang perangkat lunak agar dapat menggunakan *API* sesuai dengan keperluan berkaitan dengan parameter data yang dibutuhkan serta data keluaran yang diterima.[26]

2.8.1 Gmaps

Google Maps (Gmaps) merupakan layanan yang disediakan *Google* untuk membaca lokasi dari *GPS* yang ada di s*martphone* berbasis *Android*. Penggunaan *Gmaps* digunakan untuk mengetahui lokasi *smartphone* pengguna melalui fitur *Location Services* [27]. *Data* yang diberikan oleh *Location Services* direkam untuk merekam perpindahan posisi *GPS* saat joging dilakukan pengguna. Berikut menambahkan *plugin Gmaps* kedalam Aplikasi Diet dan Joging :

1. Menambahkan *plugins maps platform* dan *gms service* pada *file gradle.build.*

```
plugins {
    id 'com.google.android.libraries.mapsplatform.secrets-gradle-plugin'
    id 'com.google.gms.google-services'
}
```

2. Menambahkan *Dependency gms play-services.*

```
dependencies {
implementation 'com.google.android.gms:play-services maps:18.1.0'
implementation 'com.google.android.gms:play-services loc tion:21.0.1'
}
```

3. Menambahkan *plugins maps platform* didalam file Project Gradle.

```
plugins {
    id 'com.google.android.libraries.mapsplatform.secrets-gradle plugin' version
'2.0.1' apply false
  }
```

Location Services merupakan fitur yang disediakan oleh *Gmaps* untuk memberikan posisi terkini dari ponsel pengguna [27]. Hal ini dapat dilakukan dengan menggunakan pengkodean sebagai berikut:

1. Import paket *Location Services*

   Import paket *Location Services* berguna untuk mengakses seluruh fitur yang terdapat pada *Location Services*.

```
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationCallback
import com.google.android.gms.location.LocationRequest.PRIORITY_HIGH_ACCURACY
import com.google.android.gms.location.LocationResult
import com.google.android.gms.location.LocationServices
```

2. Implementasi *Location Services*

   Implementasi *Location Services* dapat dilakukan dengan menggunakan fungsi *locationRequest*(). Fungsi tersebut digunakan untuk mendapatkan lokasi terkni dari ponsel pengguna. Ada beberapa parameter yang harus disiapkan seperti *Interval, fastestInterval,* dan *priority.*

```
fusedLocationProviderClient = LocationServices.getFusedLocationProviderClient(this)
val request = com.google.android.gms.location.LocationRequest().apply {
    interval = LOCATION_UPDATE_INTERVAL
    fastestInterval = FASTEST_LOCATION_INTERVAL
    priority = PRIORITY_HIGH_ACCURACY
}
fusedLocationProviderClient.requestLocationUpdates(
    request,
    locationCallback,
    Looper.getMainLooper()
)
```

3. Implementasi fungsi *locationCallback()*

   Fungsi *locationCallback()* digunakan untuk menjalankan sebuah fungsi setelah lokasi *GPS* didapat dan titik *GPS* yang didapat akan direkam oleh aplikasi untuk mengetahui rute yang ditempuh oleh pengguna. Berikut kode fungsi *locationCallback() :*

```
val locationCallback = object: LocationCallback() {
    override fun onLocationResult(p0: LocationResult) {
        super.onLocationResult(p0)
        if(isTracking.value!!) {
            p0?.locations?.let { locations ->
                for(location in locations) {
```

```
                addPathPoint(location)
                Timber.d("New Location: ${location.latitude} ${location.longitude}")
            }
        }
    }
}
}
```

2.8.2 Low Carb Recipes API

Low Carb Recipes Api adalah layanan yang berguna untuk menyediakan *Food Suggestion*. *API* tersebut digunakan untuk menyajikan *Food Suggestion* bagi pengguna. *Low Carb Recipes API* memiliki beberapa *field* yang dapat digunakan sebagai parameter pada *endpoint usage*. Beberapa parameter tersebut adalah sebagai berikut:

Endpoint Usage

Endpoint Usage adalah adalah titik komunikasi dalam sistem komputer atau aplikasi yang menerima permintaan dan mengirimkan respons. Dalam konteks API, endpoint adalah URL atau path yang diakses oleh klien untuk berinteraksi dengan layanan yang disediakan oleh API. Penggunaan *Endpoint Low Carb Recipes API* sebagai berikut:

1. *Base URL*

Base URL adalah *URL* yang digunakan untuk terhubung dengan pihak *API*. *Base URL* dari *Low Carb Recipes API* sebagai berikut:

https://low-carb-recipes.p.rapidapi.com/

2. *Search Parameter*

Search Parameter digunakan untuk memberikan data masukan kepada *API* berupa maksimum kalori yang terkandung di dalam sebuah makanan beserta jenis makanan yang ditulis dalam *tags* dan *maxCalories* diisi dengan kebutuhan kalori pengguna. Berikut *Endpoint Usage* dari *Low Carb Recipes Api*.

https://low-carb-recipes.p.rapidapi.com/search?tags=low-carb&maxCalories=

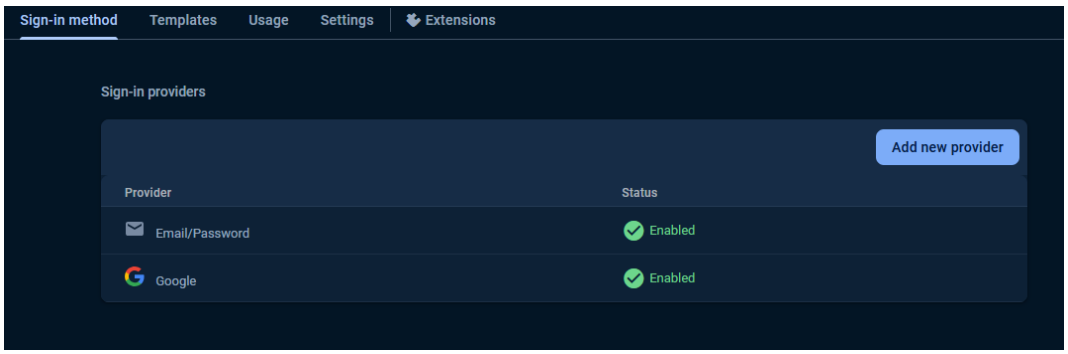
2.9 *Firebase*

Firebase menyediakan *NoSQL Database* serta beberapa fitur di antaranya adalah *Authentication* untuk sistem *login* dari *Firebase Authentication*, *Cloud Firestore* untuk menyimpan seluruh kebutuhan data objek dengan *NoSQL*

Database, serta *Firebase Storage* untuk menyimpan segala jenis kebutuhan data seperti suara, gambar dan video. *Firebase* dapat digunakan melalui *Library* yang tersedia bukan melalui *HTTP request*.[28]

2.9.1 Firebase Authentication

Firebase Authentication adalah *web service* yang digunakan untuk melakukan proses *register account, login,* dan *logout*. Tidak hanya proses *login*, *Firebase Authentication* juga menyimpan data pengguna dalam *Cloud Firestore. Firebase Authentication* juga dapat menyajikan data apakah pengguna sudah dalam kondisi *login* atau belum sehingga aplikasi dapat memberikan hak akses layanan yang baik kepada pengguna yang sudah *login* [28]. Fitur yang digunakan hanya *login* dengan *email* dan *password* saja agar lebih mudah dalam proses pengembangan. Hal ini dapat dilihat pada Gambar 2.1.



Gambar 2.1 *Firebase Authentication*

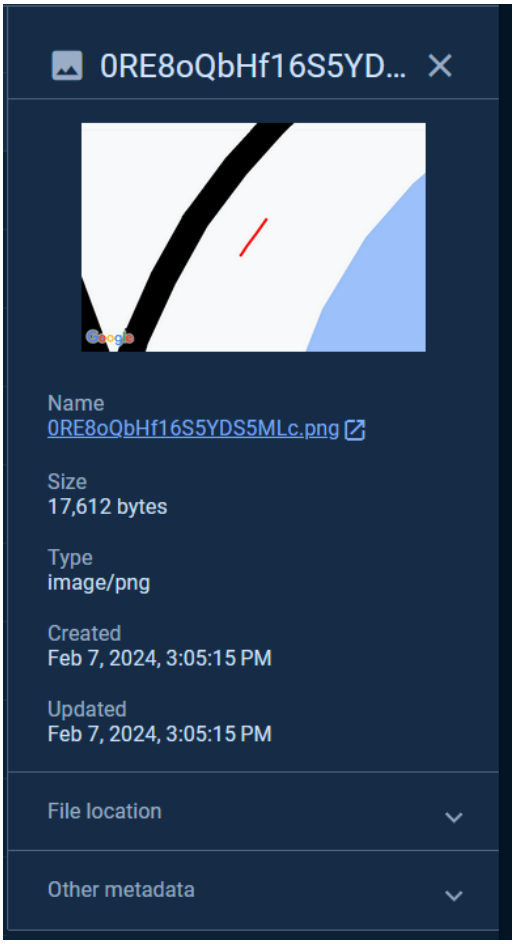
2.9.2 Cloud Firestore

Cloud Firestore merupakan *online object database* yang digunakan untuk menyimpan data objek yang diperlukan. *Cloud Firestore* juga menyediakan fungsi untuk menyimpan data objek yang ada di ponsel pengguna ke dalam *Cloud Firestore* dengan mudah [28]. Berikut contoh data yang disimpan didalam *Cloud Firestore* pada Gambar 2.2.

Gambar 2.2 *Cloud Firestore*

2.9.3 Cloud Storage

Cloud Storage merupakan suatu layanan *Firebase* yang digunakan sebagai tempat penyimpanan data berupa foto, *video*, *text*, atau audio secara *online*. *Cloud Storage* menyediakan link agar data dapat diakses. *Cloud Storage* digunakan untuk menyimpan gambar rute joging dan menyimpan foto profil [28]. Berikut contoh foto yang disimpan pada *Cloud Storage* pada Gambar 2.3.



Gambar 2.3 *Cloud Storage*

2.9.4 ML Kit Translation

ML Kit Translation adalah fitur yang disediakan oleh android untuk melakukan translasi antar bahasa. Translasi bahasa membuat informasi yang didapati dalam bahasa Inggris dapat dirubah menjadi bahasa Indonesia menggunakan fitur *Translation* yang disediakan oleh *ML Kit Translation.* Berikut tahap persiapan sebelum menggunakan *ML Kit Translation* :

1. Menambahkan modul *ML Kit Translation*

```
dependencies {
 implementation 'com.google.mlkit:translate:17.0.2'
}
```

2. Membuat *Translator Options*

```
// Create an English-German translator:
val options = TranslatorOptions.Builder()
```

```
    .setSourceLanguage(TranslateLanguage.ENGLISH)
    .setTargetLanguage(TranslateLanguage.GERMAN)
    .build()
val englishGermanTranslator = Translation.getClient(options)
```

3. Memastikan bahwa model bahasa yang digunakan tersedia

```
var conditions = DownloadConditions.Builder()
    .requireWifi()
    .build()
englishGermanTranslator.downloadModelIfNeeded(conditions)
    .addOnSuccessListener {
        // Model downloaded successfully. Okay to start translating.
        // (Set a flag, unhide the translation UI, etc.)
    }
    .addOnFailureListener { exception ->
        // Model couldn't be downloaded or other internal error.
        // ...
    }
```

4. Menjalankan fungsi *translate()*

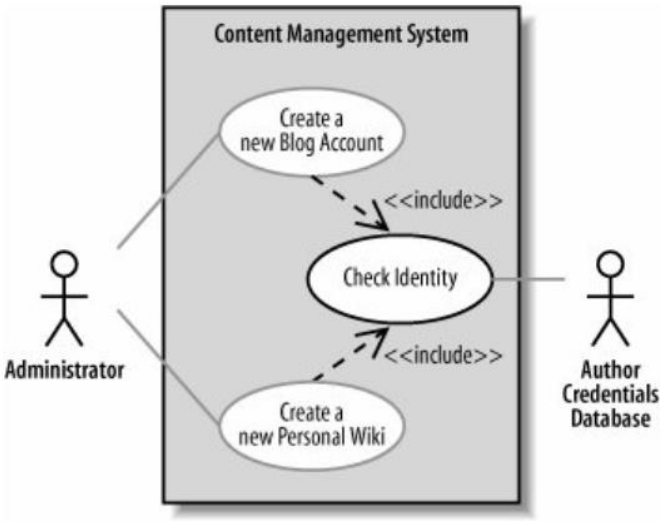
```
englishGermanTranslator.translate(text)
    .addOnSuccessListener { translatedText ->
        // Translation successful.
    }
    .addOnFailureListener { exception ->
        // Error.
        // ...
    }
```

2.10 *Use Case Diagram*

Use Case Diagram merupakan salah satu diagram yang digunakan untuk memodelkan aspek perilaku sebuah sistem informasi. *Use Case Diagram* biasa digunakan untuk memvisualisasikan, menspesifikasikan, serta mendokumentasikan kebutuhan perilaku sistem. Tujuan dari pemodelan *Use Case* adalah memodelkan perilaku pengguna dan sistem dalam satu kesatuan, Sehingga sistem dapat digambarkan secara menyeluruh [29], [30]. Tujuan utama dari pemodelan *Use Case* sebagai berikut:

1. Untuk membantu memutuskan dan mendeskripsikan kebutuhan fungsional pada sistem yang dibangun.
2. Untuk menyediakan basis dalam melakukan pengujian sistem untuk menguji apakah sistem telah berjalan sesuai dengan fungsionalitas yang diminta.

3. Untuk menyediakan gambaran besar guna merinci kelas-kelas serta operasi-operasi aktual di dalam sistem. Berikut contoh gambar *Use Case Diagram* pada Gambar 2.4.



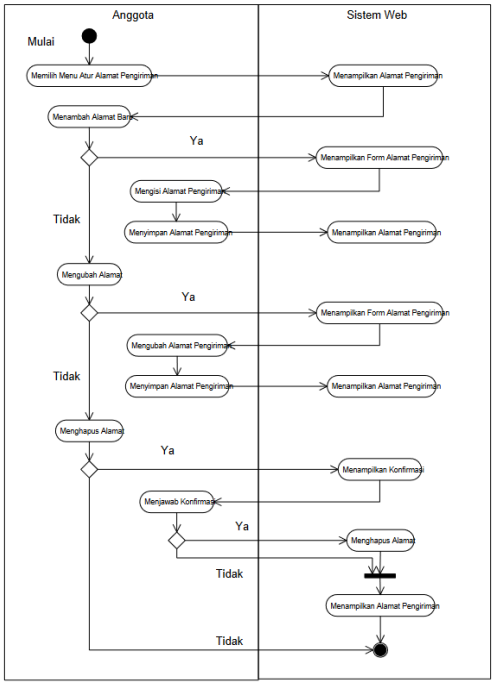
Gambar 2.4 Contoh *Use Case Diagram*

2.11 *Activity Diagram*

Activity Diagram adalah diagram untuk mengilustrasikan kegiatan utama dan hubungan di antara kegiatan dalam suatu proses antara pengguna dengan sistem atau layanan sehingga aktivitas pengguna dapat didesain melalui *Activity Diagram* [29], [30]. *Activity Diagram* digunakan untuk beberapa hal sebagai berikut:

1. Sebagai pandangan dalam yang dilakukan di operasi.
2. Sebagai pandangan dalam bagaimana objek–objek dapat bekerja.
3. Sebagai pandangan dalam aksi–aksi dan juga bagaimana pengaruhnya pada objek–objek.
4. Sebagai logika dari proses bisnis.

Contoh gambar *Activity Diagram* ditunjukkan pada Gambar 2.5.



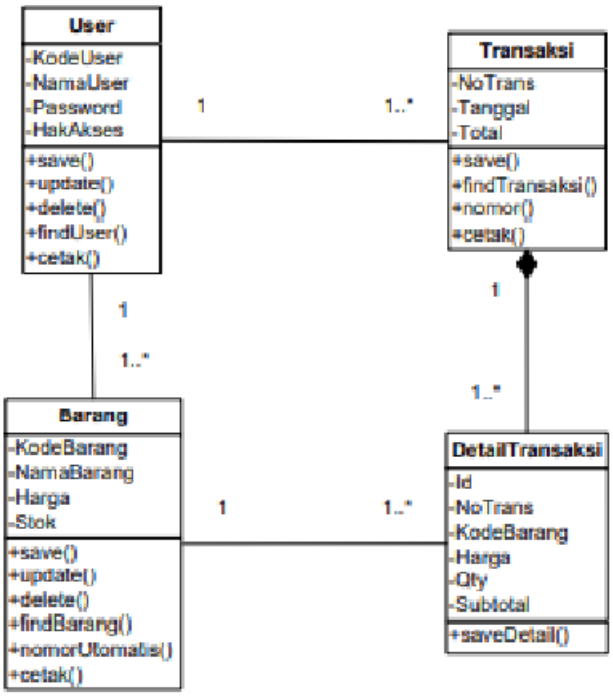
Gambar 2.5 Contoh *Activity Diagram*

2.12 *Class Diagram*

Class Diagram merupakan diagram yang dapat digunakan untuk memodelkan aspek perilaku sistem. *Class Diagram* dapat digunakan untuk memvisualisasikan, menspesifikasikan, dan juga mendokumentasikan kebutuhan perilaku sistem [29], [30]. Tujuan utama dari pemodelan *Class Diagram* sebagai berikut:

1. Dapat memodelkan kosa kata didalam sebuah sistem.
2. Dapat memodelkan tipe data *primitive*.
3. Dapat memodelkan entitas yang bukan perangkat lunak
4. Dapat memodelkan skema logika basis data.

Contoh *Class Diagram* ditunjukkan pada Gambar 2.6.



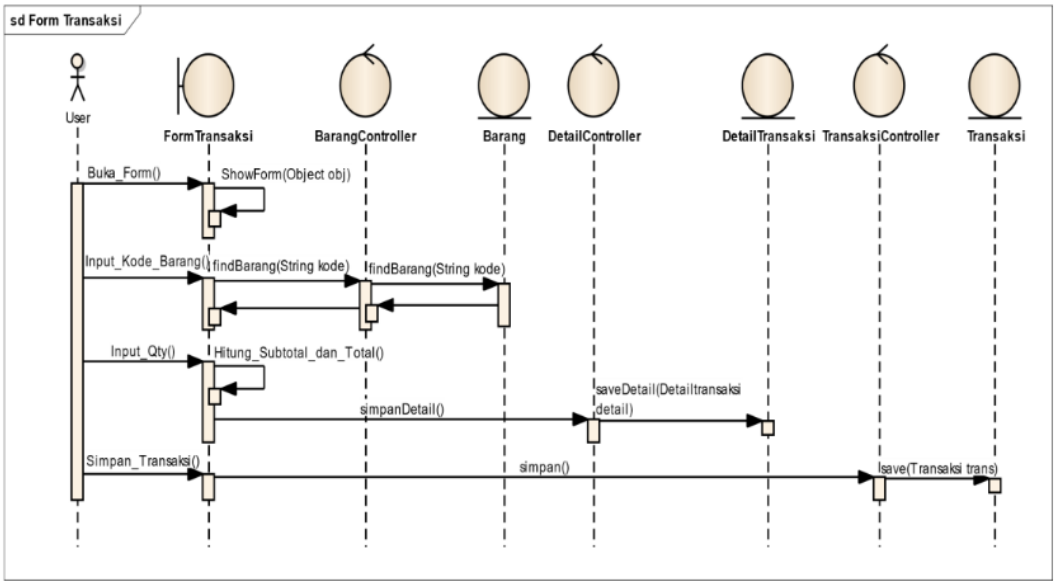
Gambar 2.6 Contoh *Class Diagram*

2.13 *Sequence Diagram*

Sequence Diagram merupakan diagram yang dapat menampilkan interaksi antara pengguna dengan aplikasi secara detail berdasarkan konsep aplikasi yang akan dibuat sehingga *developer* dapat menentukan alur program yang akan dibuat [29], [30]. *Sequence Diagram* memiliki kegunaan sebgai berikut:

1. Untuk *overview* perilaku bisnis.
2. Untuk menunjukan objek-objek yang diperlukan.
3. Untuk mendokumentasikan *scenario* dari suatu diagram *Use Case.*
4. Untuk memberikan jalur pengaksesan.

Contoh *Sequence Diagram* ditunjukkan pada Gambar 2.7 Contoh *Sequence Diagram*.



Gambar 2.7 Contoh *Sequence Diagram*