

Implementasi Steganografi dalam Penyembunyian Pesan
pada Citra Digital dengan Metode *Least Significant Bit*

Proposal

Disusun untuk melengkapi syarat-syarat
guna memperoleh gelar Sarjana Komputer



AMELIA APRILIANI

3145143626

PROGRAM STUDI ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS NEGERI JAKARTA
2018

LEMBAR PENGESAHAN

Dengan ini saya mahasiswa Fakultas Matematika dan Ilmu Pengetahuan Alam,
Universitas Negeri Jakarta

Nama : Amelia Apriliani
No. Registrasi : 3145143626
Jurusan : Ilmu Komputer
Judul : Implementasi Steganografi dalam
Penyembunyian Pesan pada Citra Digital
dengan Metode *Least Significant Bit*.

Menyatakan bahwa proposal ini telah siap diajukan untuk seminar pra skripsi.

Menyetuji,

Dosen Pembimbing I

Dosen Pembimbing II

Drs. Mulyono, M.Kom.

NIP. 119660517 199403 1 003

Ratna Widayati, S.Si, M.Kom.

NIP. 19750925 200212 2 002

Mengetahui,

Ketua Program Studi Ilmu Komputer

Drs. Mulyono, M.Kom.

NIP. 119660517 199403 1 003

DAFTAR ISI

DAFTAR ISI	3
DAFTAR GAMBAR	4
DAFTAR TABEL	5
I LATAR BELAKANG	1
1.1 Latar Belakang Masalah	1
1.2 Batasan Masalah	2
1.3 Rumusan Masalah	3
1.4 Tujuan Penelitian	3
1.5 Manfaat Penelitian	3
1.6 Jenis Penelitian	4
II KAJIAN TEORI	5
2.1 Pengembangan Aplikasi Perangkat Lunak	5
2.2 <i>Unified Modeling Language</i> (UML)	8
2.2.1 <i>Use Case Diagram</i>	8
2.2.2 <i>Class Diagram</i>	12
2.2.3 <i>Activity Diagram</i>	13
2.3 <i>Structured Query Language</i> (SQL)	14
2.4 <i>Entity Relationship Diagram</i> (ERD)	15
2.5 Pengertian Android	19
2.6 Android Studio	21
2.7 <i>Application Programming Interface</i> (API)	22
2.8 Integrasi YouTube dengan Aplikasi Android	23

III IMPLEMENTASI PROGRAM	30
3.1 Identifikasi	30
3.2 Desain	34
3.2.1 <i>Use Case Diagram</i>	36
3.2.2 <i>Entity Relationship Diagram</i>	37
3.2.3 <i>Class Diagram</i>	38
3.2.4 <i>Activity Diagram</i>	42
3.2.5 Desain <i>Wireframe</i> atau Kerangka Desain	43
3.2.6 Desain <i>Mock-Up</i> atau <i>User Interface</i>	50
3.3 Konstruksi dan Pembangunan	57
3.3.1 Membuat Basis Data	58
3.3.2 Membuat API untuk Mengakses Basis Data	58
3.3.3 Melakukan Input Data ke Basis Data	60
3.3.4 Implementasi Desain Aplikasi	62
3.3.5 Implementasi Metode Pertukaran Data antara Aplikasi dengan Basis Data	63
3.3.6 Mengintegrasikan YouTube dengan Aplikasi Android	64
3.3.7 Mengimplementasi Fitur Lainnya	66
DAFTAR PUSTAKA	70

DAFTAR GAMBAR

Gambar 2.1	Empat Tahapan Utama SDLC	6
Gambar 2.2	Model Spiral	7
Gambar 2.3	Aktor dalam <i>Use Case</i>	9
Gambar 2.4	Sistem dengan Bentuk Oval	9
Gambar 2.5	Garis Penghubungan antara Aktor dan Sistem	9
Gambar 2.6	Contoh Hubungan <i>Include</i>	10
Gambar 2.7	Contoh Hubungan <i>Extend</i>	11
Gambar 2.8	Contoh <i>Use Case</i> dalam Penggunaan Aplikasi Resep Masakan	12
Gambar 2.9	Contoh <i>Activity Diagram</i> dalam sistem Point Of Sales (POS)	13
Gambar 2.10	ERD Studi Kasus Minimarket	18
Gambar 2.11	<i>Stack Diagram</i> Hirarki Aplikasi Android dengan Komponen Lainnya	20
Gambar 2.12	Cara Kerja API	23
Gambar 2.13	Cara Kerja YouTube API	24
Gambar 2.14	Laman Unduh YouTube Player Android API	25
Gambar 2.15	Tampilan <i>tree Project</i>	25
Gambar 2.16	Menyalin Berkas YouTube API	26
Gambar 2.17	Menambah Baris Kode	26
Gambar 2.18	<i>Sync Now</i>	27
Gambar 2.19	Menambah Baris Kode pada Activity	27
Gambar 2.20	Inisialisasi YouTubePlayerView	27
Gambar 2.21	Inisialisasi YouTubePlayerView ke dalam <i>method onCreate</i> . .	27
Gambar 2.22	Inisialisasi Tampilan YouTube	28
Gambar 2.23	Implementasi <i>Method</i> YouTube	28

Gambar 2.24 Inisialisasi Tampilan YouTube	28
Gambar 2.25 Contoh Karakter Alfanumerik YouTube	29
Gambar 2.26 Contoh Pengaplikasian Karakter Alfanumerik YouTube . . .	29
Gambar 3.1 Alur Kerja Aplikasi	30
Gambar 3.2 <i>Pie Chart</i> Hasil Kuisioner Ahli	31
Gambar 3.3 <i>Pie Chart</i> Hasil Kuisioner Awam	32
Gambar 3.4 Contoh <i>Wireframe</i>	35
Gambar 3.5 Contoh <i>Mock-Up</i>	35
Gambar 3.6 Desain <i>ERD</i> Aplikasi yang Dikembangkan Penulis	37
Gambar 3.7 <i>Class Diagram</i> Menu Utama	39
Gambar 3.8 <i>Class Diagram</i> Detail Resep	40
Gambar 3.9 <i>Class Diagram</i> Wishlist	41
Gambar 3.10 <i>Activity Diagram</i> dari Aplikasi yang Dibuat Penulis	42
Gambar 3.11 Halaman Awal	43
Gambar 3.12 Menu Utama	44
Gambar 3.13 Navigasi	45
Gambar 3.14 Menu Utama Tab Rekomendasi	45
Gambar 3.15 Menu Utama Tab Jenis	46
Gambar 3.16 Detail Resep	47
Gambar 3.17 Keranjang atau Wishlist	47
Gambar 3.18 Total Bahan	48
Gambar 3.19 Total Harga	48
Gambar 3.20 Rincian Total Harga	49
Gambar 3.21 Hasil Pencarian	49
Gambar 3.22 Halaman Beri Masukan	50
Gambar 3.23 Halaman Awal atau Halaman Selamat Datang	51

Gambar 3.24 Halaman Utama	51
Gambar 3.25 Menu Navigasi	52
Gambar 3.26 Halaman Utama <i>Tab</i> Rekomendasi	52
Gambar 3.27 Halaman Utama <i>Tab</i> Jenis	53
Gambar 3.28 Detail Bahan Resep	53
Gambar 3.29 Detail Cara Memasak	54
Gambar 3.30 <i>Wishlist</i>	54
Gambar 3.31 Total Bahan	55
Gambar 3.32 Total Harga	55
Gambar 3.33 Rincian Total Harga	56
Gambar 3.34 <i>Tab</i> Pencarian	56
Gambar 3.35 Laman Beri Masukan atau <i>Feedback</i>	57
Gambar 3.36 Basis Data yang Dibuat dengan phpMyAdmin	58
Gambar 3.37 Bagan Alur Kerja MVC	59
Gambar 3.38 Bagan Alur Kerja CodeIgniter-Based API	59
Gambar 3.39 Sampel Kode CodeIgniter API-Based yang Dibuat Penulis	60
Gambar 3.40 Input Data dengan phpMyAdmin	61
Gambar 3.41 Tampilan List Resep pada Aplikasi Berbasis Web	61
Gambar 3.42 Tampilan Input Resep pada Aplikasi Berbasis Web	62
Gambar 3.43 Bagan Alur Kerja CodeIgniter-Based API Hasil Modifikasi	62
Gambar 3.44 Bagan Alur Kerja Form GET/POST	63
Gambar 3.45 Bagan Alur Kerja Volley	63
Gambar 3.46 Bagan Alur Kerja YouTubePlayerView pada sebuah Activity	64
Gambar 3.47 Bagan Alur Kerja YouTubePlayerFragment pada sebuah Activity	65

Gambar 3.48 Sampel Kode Implementasi YouTubePlayerFragment pada Metode onCreate sebuah Activity	66
Gambar 3.49 Alur Penyimpanan pada SQLite dan Pengolahan Total Bahan dan Harga Bahan	67

DAFTAR TABEL

Tabel 2.1	Komponen-komponen Pembentuk ERD	16
-----------	---	----

BAB I

LATAR BELAKANG

1.1 Latar Belakang Masalah

Saat ini *internet* sudah berkembang menjadi salah satu media yang sangat populer di berbagai dunia (Bunyamin;Adrian, 2009). Perkembangan *internet* memberikan pengaruh besar terhadap kemudahan dalam berkomunikasi dan menyampaikan informasi. Komunikasi merupakan salah satu hal yang penting bagi manusia. Manusia yang merupakan makhluk sosial cenderung melakukan komunikasi setiap hari, baik secara langsung maupun melalui media elektronik. Manusia melakukan komunikasi untuk bertukar informasi.

Kemudahan dalam berkomunikasi memberikan dampak positif dan negatif. Dampak positifnya yaitu cepatnya informasi dapat tersebar, baik antar daerah maupun antar negara. Dan dampak negatifnya adalah semakin berkembangnya kejahatan dalam penggunaan informasi. Dengan berbagai teknik, banyak orang yang mencoba untuk mengakses informasi yang bukan haknya. Maka dari itu harus berkembang juga pengamanan sistem informasi.

Teknik pengamanan informasi yang ada saat ini seperti kriptografi dan steganografi. Kriptografi adalah suatu ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikan ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Kriptografi telah ada dan digunakan sejak berabad-abad yang lalu dikenal dengan istilah kriptografi klasik, yang bekerja pada mode karakter alfabet (Rakhmat;Fairuzabadi, 2010).

Steganografi adalah seni dan sains komunikasi pesan yang tak terlihat. Hal ini dilakukan dengan menyembunyikan informasi dalam informasi lain, misalnya me-

nyembunyikan keberadaan informasi yang dikomunikasikan. Kata steganografi berasal dari kata Yunani "stegos" yang berarti "cover" dan "grafia" yang berarti "menulis" yang mendefinisikannya sebagai "tulisan tertutup" (Mrs., Kadam, Koshti;Dunghav, 2012).

Salah satu metode steganografi adalah *Least Significant Bit* (LSB). Algoritma LSB, menggantikan bit paling signifikan pada *file cover* sesuai dengan bit pesan. Teknik ini adalah teknik yang paling populer digunakan dalam steganografi untuk menyembunyikan pesan. Teknik ini biasanya efektif, karena substitusi LSB tidak menyebabkan degradasi kualitas yang signifikan (Joshi;Yadav, 2015).

1.2 Batasan Masalah

Batasan masalah dalam tugas akhir ini mencakup:

- *Software* yang digunakan adalah Matlab R2013b.
- Peneliti akan langsung menggunakan *smartphone* berbasis Android dalam proses *debugging* dan *testing*. Versi android yang digunakan adalah Lollipop (Android 5.0) dengan API 21.
- Format *file* citra *digital* yang dapat digunakan untuk menyimpan pesan adalah berformat *.bmp.
- Format *file* citra *digital* yang dihasilkan dari program steganografi ini adalah berformat *.bmp.
- Pesan yang dapat disimpan hanya berformat *.txt.
- Metode yang digunakan adalah *Least Significant Bit*.

1.3 Rumusan Masalah

Rumusan masalah berdasarkan latar belakang di atas adalah:

1. Bagaimana cara menyembunyikan teks dalam proses steganografi dengan menggunakan metode *Least Significant Bit*?
2. Bagaimana perubahan dalam *file* citra hasil keluaran sebelum dan sesudah disipkan pesan teks?

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Memberikan informasi bagaimana teknik steganografi dapat diterapkan untuk menyembunyikan teks dalam *file* citra *digital* dengan menggunakan metode *Least Significant Bit*.
2. Mengetahui perubahan yang terjadi dari hasil keluaran *file* citra *digital*.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat sebagai berikut:

1. Bagi Penulis, diharapkan dapat menambah pengetahuan dan pemahaman tentang steganografi.
2. Bagi Program Studi Ilmu Komputer, Penulisan penelitian ini memberikan gambaran bagi seluruh mahasiswa khususnya bagi mahasiswa program studi Ilmu Komputer Universitas Negeri Jakarta tentang bagaimana teknik steganografi dapat menyembunyikan pesan dalam *file* citra *digital*.

1.6 Jenis Penelitian

Jenis Penelitian yang dijalani oleh Peneliti berjenis ... Jenis penelitian ini mengarahkan penulis kepada ...

BAB II

KAJIAN TEORI

Kumpulan resep masakan pada umumnya memuat berbagai macam makanan yang siap untuk dimasak. Makanan merupakan elemen penting yang membentuk suatu resep karena makanan adalah produk hasil akhir yang ingin dicapai melalui panduan yang terdapat dalam resep masakan. Resep masakan kemudian dijadikan konten utama dalam pengembangan yang dilakukan oleh penulis dengan menggunakan Android sebagai wadah bagi aplikasi atau perangkat lunak tersebut. Pengembangan resep masakan ini dilakukan berdasarkan tahapan *System Development Life Cycle* (SDLC) yang berlaku secara umum dalam dunia *software developing*.

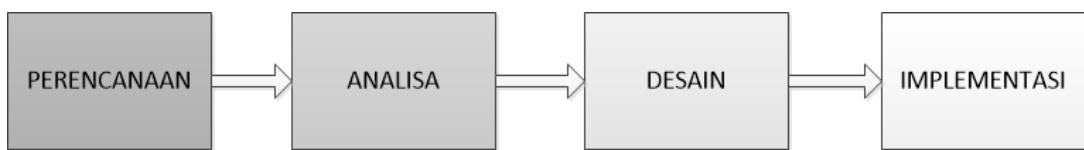
2.1 Pengembangan Aplikasi Perangkat Lunak

Pengembangan perangkat lunak pada umumnya sama dengan membangun sebuah gedung. Proses pertama dimulai dengan sebuah ide atau gagasan, lalu dilanjutkan dengan merubah ide atau gagasan tersebut menjadi sebuah desain sehingga dapat memberi gambaran lebih jelas tentang ide tersebut. Proses selanjutnya bersifat opsional, yakni melakukan revisi atau perubahan terhadap desain apabila dirasa tidak sesuai dengan yang diinginkan. Setelah itu, dilanjutkan dengan melakukan pembuatan cetak biru terhadap desain tersebut sehingga memperoleh gambaran yang lebih mendetail lagi dari masing-masing bagian gedung tersebut. Proses yang terakhir adalah membangun gedung tersebut berdasarkan cetak biru yang ada [9].

Dalam proses pengembangan aplikasi perangkat lunak, terdapat istilah SDLC atau *System Development Life Cycle*. SDLC adalah sebuah metode atau proses untuk memberikan pengertian bagaimana sebuah sistem informasi dapat mendukung kebutuhan bisnis dengan melakukan desain sebuah sistem, membangun sistem tersebut,

dan mengimplementasikannya ke pengguna [7].

SDLC memiliki 4 tahapan utama yakni perencanaan, analisa, desain, dan implementasi. Dalam sebuah proyek pengembangan aplikasi perangkat lunak pastilah memiliki keempat tahapan tersebut, namun cara untuk melaksanakan tahapan-tahapan tersebut dapat berbeda-beda antara satu proyek dengan proyek lainnya. Setiap tahapan tersebut memiliki langkah-langkah tersendiri dalam menjalankan tahapan-tahapan tersebut sesuai dengan tujuan akhir dari masing-masing tahapan tersebut.



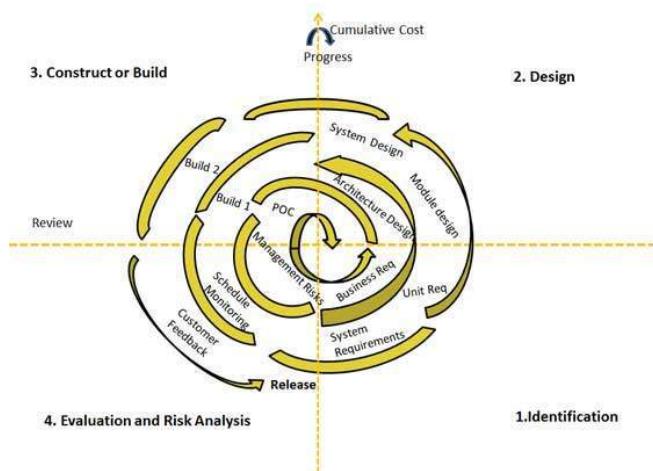
Gambar 2.1: Empat Tahapan Utama SDLC

Jenis-jenis pelaksanaan dari tahapan-tahapan SDLC terdiri dari [15]:

1. *Linear Process Flow*, yang mengeksekusi 4 tahapan tersebut secara berurutan
2. *Iterative Process Flow*, yang mengeksekusi secara berulang satu atau beberapa tahapan sebelum melanjutkan ke tahapan selanjutnya
3. *Evolutionary Process Flow*, mengeksekusi tahapan tersebut secara melingkar
4. *Parallel Process Flow*, mengeksekusi satu atau beberapa tahapan secara bersamaan dengan tahapan lain.

Dalam metode SDLC terdapat beberapa model pengembangan yang dapat dijadikan acuan dasar kerangka kerja dalam proses pengembangan sebuah aplikasi perangkat lunak, yakni *Waterfall Model*, *Iterative Model*, *Spiral Model*, *V-Model*, *Big Bang Model*, *Agile Model*, *Rapid Application Development (RAD) Model*, dan *Prototyping Model*. Dalam pengembangan aplikasi resep masakan ini, penulis memilih menggunakan *Spiral Model*.

Model spiral memadukan sistematika dari pembangunan antara model iteratif (*Iterative Model*) dengan aspek pengendalian dari model *waterfall* (*Waterfall Model*). Model spiral termasuk dalam jenis *Evolutionary Process Flow*, yang mengeksekusi tahapan pelaksanaan secara melingkar [3]. Tahapan-tahapan yang akan dilakukan dalam penelitian ini mengikuti model pengembangan spiral, yaitu: *Identification* (identifikasi kebutuhan), *Design* (Desain Sistem), *Construct and Build* (Pembuatan Sistem), dan *Evaluation (delivery feedback and release)*



Gambar 2.2: Model Spiral

Penulis memilih menggunakan model spiral sebagai model dari SDLC dalam proses pengembangan aplikasi kumpulan resep masakan terkoneksi *food channel* YouTube dengan alasan apabila dalam proses pengembangan terdapat kesalahan, baik dalam proses identifikasi, desain, atau pembuatan sistem, maka penulis dapat kembali ke tahap dimana ditemukan kesalahan untuk kemudian memperbaiki kesalahan tersebut sehingga proses pengembangan dapat dilaksanakan secara semestinya dan penulis juga tidak melanggar ketentuan yang telah ditetapkan oleh metode pengembangan SDLC karena fleksibilitas dari model spiral itu sendiri.

2.2 Unified Modeling Language (UML)

Unified Modeling Language atau UML adalah sebuah Bahasa pemodelan terstandarisasi dan digunakan untuk berbagai kebutuhan dalam memodelkan pembuatan *software* yang berorientasi pada obyek (*object-oriented*). Standar tersebut dibuat dan dikelola oleh Object Management Group (OMG) pada tahun 1997 dan telah menjadi standar di bidang industri pembuatan *software*. UML meliputi sebuah set dari teknik notasi grafis untuk membuat model dari sebuah sistem perangkat lunak berorientasi obyek secara visual. UML juga merupakan peralatan untuk menspesifikasikan dan memvisualisasikan sebuah sistem pada perangkat lunak. Hal tersebut mencakup tipe diagram yang telah terstandarisasi untuk mendeskripsikan serta memetakan sebuah aplikasi komputer atau desain serta struktur dari sebuah sistem basis data. UML memiliki fungsi untuk mengelola sistem yang besar dan kompleks. Memiliki struktur aplikasi yang jelas dapat membantu *developer* dalam mengenalkan proyek kepada orang-orang baru maupun awam [10].

Jenis UML yang digunakan oleh peneliti pada pengembangan aplikasi ini adalah *Use Case Diagram*, *Class Diagram*, dan *Activity Diagram*.

2.2.1 Use Case Diagram

Fungsionalitas dari sebuah sistem basis data maupun sebuah perangkat lunak komputer dapat diilustrasikan dengan diagram *use case*. Tugas utama dari sebuah *use case* adalah memvisualisasikan kebutuhan fungsionalitas dari sebuah sistem, termasuk hubungan antara aktor (orang yang akan berinteraksi dengan sistem) dengan proses-proses yang esensial dalam sebuah sistem untuk mencapai suatu tujuan. Hal tersebut dapat dikategorikan sebagai kumpulan langkah-langkah yang mendefinisikan interaksi antara aktor dan sistem [5].

Use Case Diagram terdiri dari beberapa komponen, yakni:

1. Aktor

Menggambarkan seorang aktor atau pengguna sistem yang akan berinteraksi dengan sistem



Gambar 2.3: Aktor dalam *Use Case*

2. Oval

Bentuk Oval menggambarkan sistem yang dapat berinteraksi dengan aktor.

Bentuk Oval tersebut biasanya disertai dengan nama aktivitas sistem di dalamnya yang terletak ditengah-tengah bentuk oval tersebut



Gambar 2.4: Sistem dengan Bentuk Oval

3. Garis Penghubung Aktor dan Sistem

Untuk menggambarkan interaksi antara Aktor dan Sistem, diperlukan sebuah garis yang menghubungkan antara aktor dan sistem. Sebuah garis digunakan untuk menghubungkan keduanya

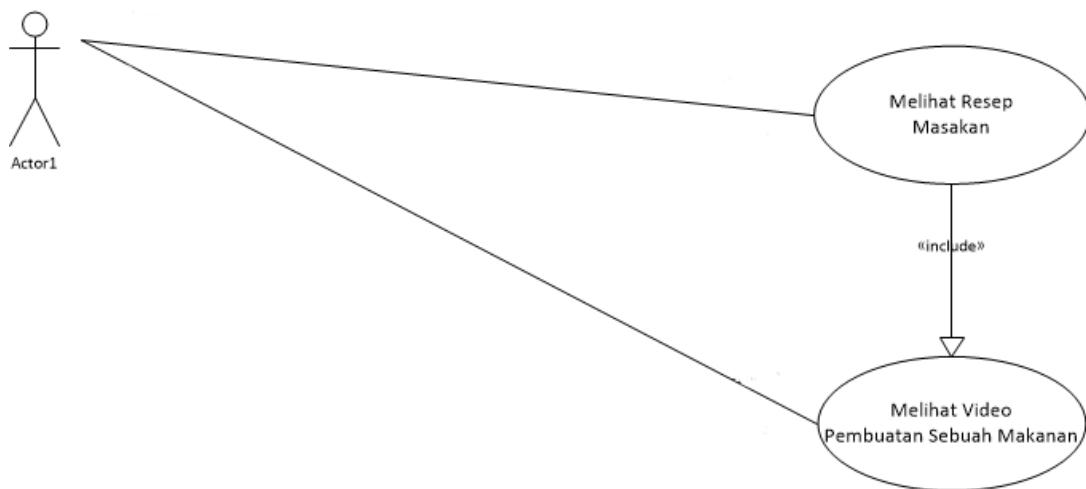


Gambar 2.5: Garis Penghubungan antara Aktor dan Sistem

Gambar 2.5 memperlihatkan garis yang menyambungkan aktor dengan interaksi sistem menyatakan bahwa Aktor dapat Melihat Resep Masakan

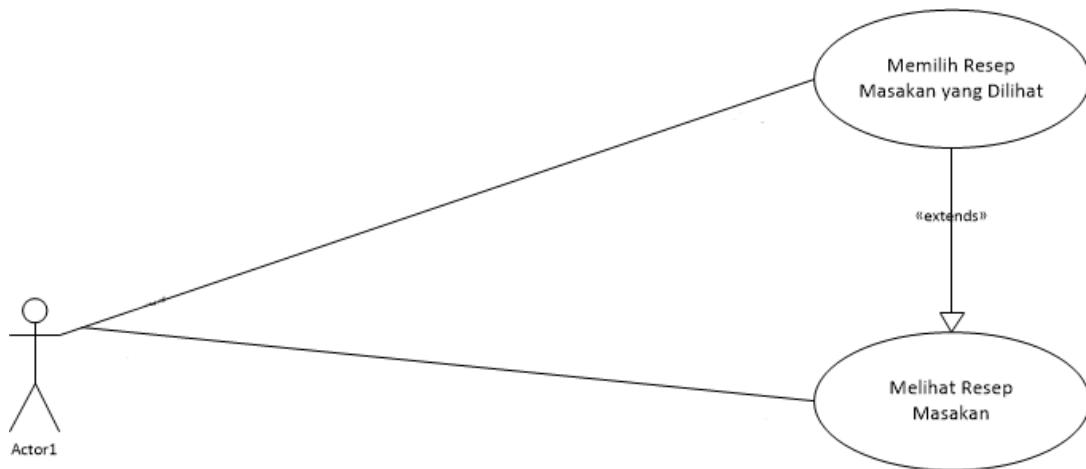
4. Garis Penghubung Antar Sistem

Sistem yang terdapat pada *use case* dapat berinteraksi satu sama lain. Terdapat dua jenis hubungan antar sistem, yakni *include* dan *extend*. Sesuai dengan namanya, *include* menyatakan bahwa sebuah interaksi sistem yang terhubung adalah bagian dari interaksi sistem lainnya



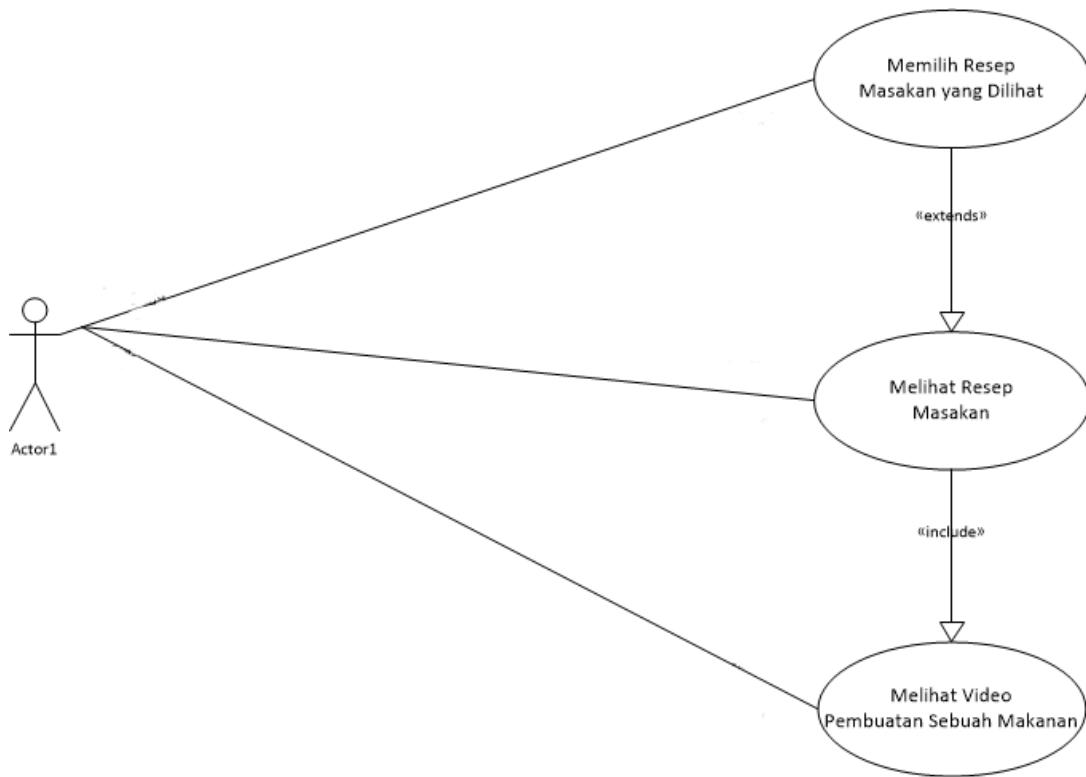
Gambar 2.6: Contoh Hubungan *Include*

Gambar 2.6 menjelaskan bahwa Melihat Video Pembuatan Sebuah Makanan merupakan bagian dari Melihat Resep Masakan. Sedangkan penghubung *extend* menyatakan bahwa interaksi sebuah sistem merupakan perluasan dari interaksi sistem lainnya



Gambar 2.7: Contoh Hubungan *Extend*

Gambar 2.7 menjelaskan bahwa Melihat Resep Masakan merupakan perluasan dari Memilih Resep Masakan yang Dilihat. Sebagai contoh, seorang Aktor dapat berinteraksi dengan sebuah sistem berbentuk aplikasi resep masakan. Interaksi yang dimungkinkan oleh sistem adalah Memilih Resep Masakan, Melihat Resep Masakan, dan Melihat Video Pembuatan Sebuah Resep Masakan. *Use Case* dari interaksi tersebut terdapat pada Gambar 2.8. Gambar tersebut menjelaskan bahwa aktor dapat memilih resep masakan dan kemudian juga melihat resep masakan yang telah dipilih. Melihat resep masakan yang dipilih merupakan ekstensi dari interaksi antara akor dengan sistem yakni memilih resep masakan yang dilihat. Kemudian pengguna juga dapat melihat video pembuatan sebuah makanan ketika melihat resep masakan. Hal tersebut dimungkinkan karena interaksi melihat resep masakan memiliki hubungan *include* dengan interaksi melihat video pembuatan sebuah makanan.



Gambar 2.8: Contoh *Use Case* dalam Penggunaan Aplikasi Resep Masakan

Jadi, dapat disimpulkan bahwa *Use Case Diagram* adalah sebuah diagram yang menggambarkan interaksi antara Aktor (pengguna) dengan sistem yang digunakan oleh Aktor.

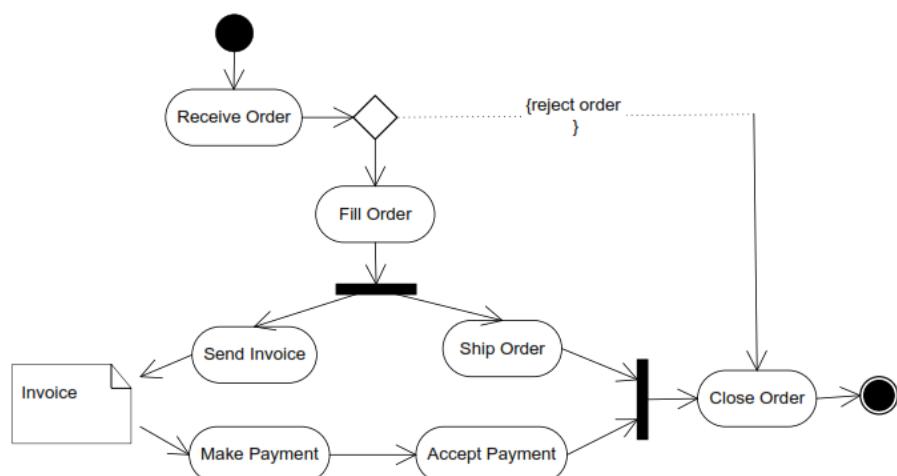
2.2.2 *Class Diagram*

Struktur statis dari sebuah aplikasi komputer atau pangkalan basis data dapat digambarkan melalui sebuah *class diagram*. *Class diagram* juga menunjukkan perbedaan entitas (pengguna, alat, maupun data) yang berelasi satu sama lain. Diagram tersebut biasa digunakan untuk menampilkan class-class secara logikal, dan untuk mengimplementasikan class yang telah dibuat. *Class diagram* merupakan unsur terpenting dalam pemodelan aplikasi berbasis obyek (*object-oriented*) dan merupakan sebuah tipe struktur diagram statis yang mendeskripsikan struktur dari sebuah sis-

tem dengan menampilkan *class* dari sistem, atribut, operasi (method), dan relasi antar *class* [10].

2.2.3 Activity Diagram

Kontrol alur serta prosedur antara dua atau lebih obyek *class* dalam memproses sebuah aktivitas dapat ditunjukkan melalui sebuah *activity diagram*. Diagram tersebut dapat digunakan untuk memodelkan proses bisnis tingkat atas dalam sebuah level unit bisnis, atau untuk memodelkan aksi-aksi dalam class internal yang bersifat *low level* (Bell, 2003). *Activity Diagram* adalah representasi grafis dari sebuah alur penggerjaan yang berupa tahap-tahap dari berbagai aktivitas serta aksi dengan dukungan pilihan, iterasi, dan persetujuan. Pada UML, *activity diagram* dapat digunakan untuk mendeskripsikan bisnis serta langkah-langkah operasional serta alur penggerjaan dari berbagai komponen yang berada dalam sebuah sistem. Sebuah *activity diagram* menjelaskan keseluruhan alur dari sebuah sistem [10]. Contoh dibawah ini menjelaskan aktivitas pemrosesan pesanan dalam sebuah sistem Point of Sales (POS).



Gambar 2.9: Contoh *Activity Diagram* dalam sistem Point Of Sales (POS)

2.3 Structured Query Language (SQL)

SQL merupakan singkatan dari *Structure Query Language*, didefinisikan sebagai suatu sintaks perintah-perintah tertentu atau bahasa program yang digunakan untuk mengelola suatu *database* [1]. SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Keandalan suatu sistem *database* (DBMS) dapat diketahui dari cara kerja *optimizer*-nya dalam melakukannya proses perintah-perintah SQL, yang dibuat oleh user maupun program-program aplikasinya. Query dikirimkan ke database dalam bentuk SQL Query Beberapa perintah yang umum digunakan adalah sebagai berikut:

1. CREATE : Untuk membuat tabel baru

```

1   CREATE TABLE <NAMA_TABLE>
2   |<NAMA KOLOM> <TIPE>,
3   |<NAMA KOLOM> <TIPE>,
4   | . . . . .
5   | PRIMARY KEY (<NAMA KOLOM>)
6   | FOREIGN KEY (<NAMA KOLOM>)
7   | REFERENCES <NAMA_TABLE>
8   | <NAMA KOLOM>

```

2. SELECT : Untuk mengambil *record* dari database yang memenuhi kriteria tertentu

```

1   SELECT <NAMA KOLOM>,
2       <NAMA KOLOM>,
3       .
4   FROM <NAMA TABEL>
5   WHERE <KONDISI>

```

3. INSERT : Untuk menambah *record* ke dalam suatu tabel

```

1   INSERT INTO <NAMA TABEL>
2   |<NAMA KOLOM>,
3   |     <NAMA KOLOM> )
4   |VALUES (<NILAI KOLOM>, <NILAI
5   |KOLOM>, . . . )

```

4. UPDATE : Untuk merubah isi *record* tertentu pada suatu tabel

```

1   UPDATE <NAMA TABEL>
2   SET (<NAMA KOLOM> = <NILAI
3   KOLOM>,<NAMA KOLOM> = <NILAI
4   KOLOM>, . . . )
5   WHERE <KONDISI>

```

5. DELETE : Untuk menghapus *record* pada suatu tabel

```

1   DELETE FROM <NAMA TABEL>
2   WHERE <KONDISI>

```

6. DROP : Untuk menghapus sebuah tabel

```
1   DROP <NAMA TABEL>
```

7. JOIN : Untuk menggabungkan dua atau lebih tabel

```

1   JOIN <NAMA TABEL>
2   ON <NAMA TABEL>.<ID> = <NAMA TABEL KEDUA>.<ID>

```

2.4 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) adalah sekumpulan cara atau peralatan untuk mendeskripsikan data-data atau objek-objek yang dibuat berdasarkan dan ber- asal dari dunia nyata yang disebut entitas (*entity*) serta relasi (*relationship*) antar entitas-entitas tersebut dengan menggunakan beberapa notasi [8]. Komponen-komponen pembentuk ERD dapat dilihat pada Tabel 2.1.

Notasi	Komponen	Keterangan
	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lain.
	Atribut	Properti yang dimiliki oleh suatu entitas, dimana dapat mendeskripsikan karakteristik dari entitas tersebut.
	Relasi	Menunjukkan hubungan diantara sejumlah entitas yang berbeda.
	Relasi 1 : 1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua
	Relasi 1 : N	Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berrelasi dengan banyak entitas pada himpunan entitas yang lain
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas pada himpunan entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya

Tabel 2.1: Komponen-komponen Pembentuk ERD

Sesuai dengan namanya, sebuah ERD memodelkan data sebagai entitas serta relasi (*relationship*). Entitas adalah data yang ingin disimpan. Chen (1976) dalam presentasinya mendeskripsikan entitas sebagai "sesuatu yang dapat diidentifikasi secara jelas." Jadi, entitas dapat berupa seseorang, sebuah tempat, obyek, kejadian, atau konsep yang ingin disimpan sebagai data.

Relasi (*relationship*) adalah koneksi antara entitas-entitas yang ada. Relasi sendiri memiliki kardinalitas, yaitu ukuran kasar dari jumlah entitas (satu atau lebih) yang akan terkait dengan entitas lain (atau entitas). Berdasarkan kardinalitasnya, terdapat tiga jenis relasi yang dapat ditemukan dalam entitas-entitas yang ada yaitu relasi *one-to-one* (1:1), *one-to-many* (1:M), dan *many-to-many* (M:N). Dalam relasi *one-to-one*, sebuah entitas dapat berasosiasi dengan satu entitas lainnya dan sebalik-

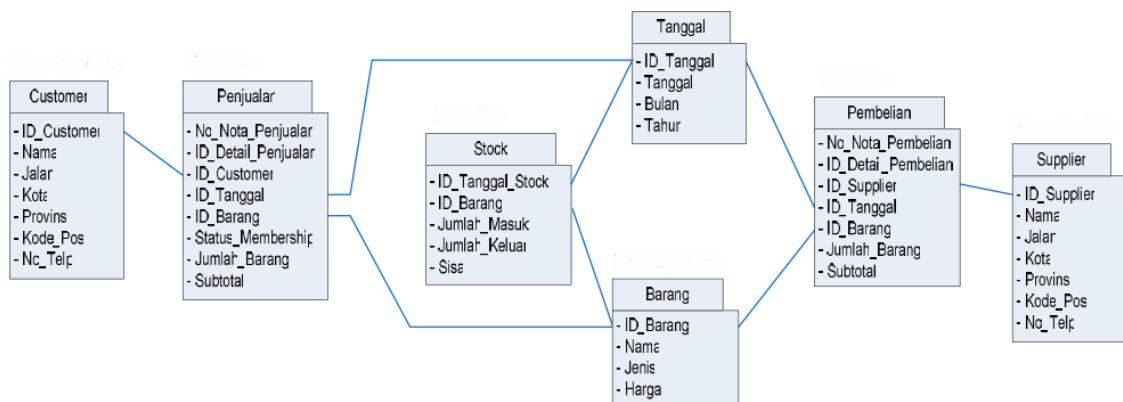
nya. Contohnya, seorang siswa hanya memiliki satu sepeda. Dengan analogi antara siswa dengan sepeda tersebut, relasi *one-to-many* menggambarkan bahwa seorang siswa dapat memiliki banyak sepeda. Sedangkan relasi *many-to-many* pada analogi siswa dengan sepeda adalah banyak siswa yang dapat memiliki banyak sepeda dan sebaliknya [4].

Atribut adalah kategori dari sebuah data yang mendeskripsikan sebuah entitas atau relasi. Contoh dari atribut adalah sebuah entitas bernama MOBIL memiliki atribut tipe, warna, id_kendaraan, dan lain sebagainya. Basis data sering digunakan untuk menyimpan data yang akan digunakan pada tahapan-tahapan selanjutnya. Sebuah atribut yang dapat digunakan untuk mencari suatu entitas tertentu disebut kunci atau *key*. Atribut yang bersifat kunci (*key*) dapat muncul secara natural apabila sebuah basis data dimodelkan dengan menggunakan ERD. Jika sebuah atribut dapat diperhitungkan menjadi sebuah ciri khas dari suatu entitas karena keunikannya, maka atribut tersebut disebut dengan *candidate key* atau kandidat kunci. *Candidate key* akan berubah menjadi *primary key* apabila telah dipilih sebagai atribut khas suatu entitas yang bersifat paling unik daripada atribut lain yang terdapat pada suatu entitas.

Studi Kasus Minimarket

Sebuah minimarket yang menjual berbagai jenis barang kebutuhan sehari-hari memiliki sebuah sistem informasi untuk mengelola penjualan secara langsung (point of sales), pengadaan barang, dan *stock control*. Proses bisnis dalam penjualan barangnya dimulai pada saat *customer* memilih barang yang akan dibeli. Setelah *customer* memutuskan untuk membeli barang tersebut, maka kasir akan meminta informasi tentang identitas *customer* untuk dicatat jika *customer* yang bersangkutan terdaftar sebagai member. Namun jika *customer* tersebut bukanlah member minimarket, maka data-data *customer* akan diabaikan. Kemudian kasir akan membuatkan nota penjualan

an barang. Setelah barang diterima oleh *customer*, *customer* akan melakukan pembayaran. Proses berakhir ketika kasir memberikan bukti pembayaran kepada *customer*. Sistem informasi yang tersedia tidak melayani proses pengembalian barang dan pesanan barang. Proses bisnis untuk pembelian barang dari supplier dimulai ketika pihak minimarket menghubungi supplier dan memesan barang. *Supplier* kemudian akan membuatkan nota pembelian. Barang yang sudah dipesan lalu akan diantarkan ke minimarket. Jika barang sudah diterima, maka proses yang terjadi adalah pembayaran dari pihak minimarket ke pihak *supplier*. Setelah semua proses pembayaran selesai, *supplier* akan memberikan bukti pembayaran dan proses selesai. Seperti halnya pada proses penjualan, proses pembelian tidak menangani pengembalian barang kepada *supplier*. Untuk proses *stock control*, dilakukan proses pencatatan terhadap barang yang di-supply, barang yang dibeli oleh *customer* dan sisa barang yang ada di gudang per harinya. Hal ini dimaksudkan agar setiap keluar masuknya barang yang ada dapat terawasi dan menjaga barang selalu tersedia di gudang. Maka ERD yang dapat dibuat dari kasus diatas adalah sebagai berikut:

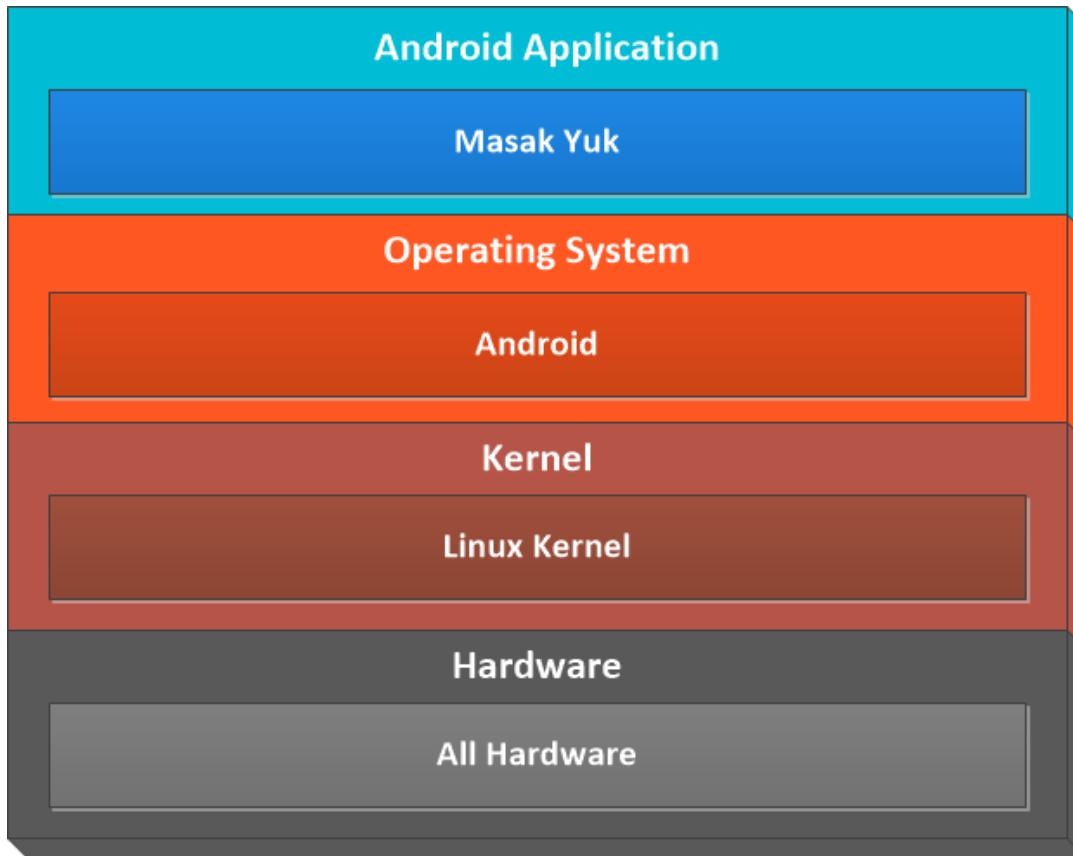


Gambar 2.10: ERD Studi Kasus Minimarket

2.5 Pengertian Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka [11]. Dikutip dari situs resmi Open Handset Alliance, yang merupakan pengembang sistem operasi Android pertama di dunia, Android dikembangkan untuk memungkinkan para pengembang aplikasi membuat suatu aplikasi *mobile* yang dapat memanfaatkan seluruh kelebihan yang ditawarkan dari suatu perangkat *mobile*. Android dibuat secara terbuka dan semua orang dapat mengembangkan dan menggunakan semua fungsionalitasnya. Contohnya, sebuah aplikasi yang dapat digunakan untuk melakukan berbagai macam fungsionalitas sebuah ponsel seperti membuat panggilan, mengirimkan pesan singkat, atau mengambil gambar dengan kamera, memungkinkan para pengembang aplikasi untuk membuat banyak sekali fitur lainnya yang dapat digunakan oleh para penggunanya. Android dibuat dengan menggunakan Kernel Linux yang bersifat terbuka (*open source*). Selain itu, Android juga merupakan hasil dari modifikasi mesin virtual (*virtual machine*) yang didesain untuk mengoptimalkan memori serta perangkat keras (*hardware*) pada sebuah ponsel. Android sendiri adalah open source yang berarti dapat terus dikembangkan menjadi sebuah teknologi baru sesuai dengan kebutuhan manusia pada saat ini. *Platform* ini akan terus berkembang seiring dengan pergerakan para pengembang aplikasi yang bekerja bersama-sama untuk membangun aplikasi *mobile* yang inovatif.

Gambar 2.11 menunjukkan hirarki aplikasi yang dibuat oleh peneliti terhadap *platform* Android.



Gambar 2.11: *Stack Diagram Hirarki Aplikasi Android dengan Komponen Lainnya*

Selain kemampuannya untuk mendukung dan memfasilitasi berbagai fungsi familiar seperti membuat panggilan, berkirim pesan elektronik, dan mencari sebuah restoran, perangkat Android dapat menjadi sebuah perangkat dengan fungsi tanpa batas dan dapat dikembangkan kapan saja bergantung pada pemikiran para pengembangnya [16]. Saat ini, Android tidak hanya terdapat pada sebuah ponsel tetapi juga tertanam dalam sebuah tablet, televisi, kacamata pintar, kulkas, mobil dan lain sebagainya. Jadi dapat disimpulkan bahwa Android merupakan sistem operasi untuk sebuah perangkat cerdas yang dapat memiliki banyak fitur dan fungsi sesuai dengan keinginan para pengembangnya untuk menjawab masalah-masalah yang terdapat dalam kehidupan manusia.

2.6 Android Studio

Android Studio merupakan sebuah *Integrated Developing Environment* (IDE) untuk *platform* Android. Android Studio ini diumumkan pada tanggal 16 Mei 2013 pada konferensi Google I/O oleh *Product Manager* Google, Ellie Powers. Android Studio bersifat *free* dibawah Apache License 2.0. Versi awal Android Studio yaitu 0.1 dirilis pada bulan Mei 2013. Kemudian dibuat versi *beta* 0.8 yang dirilis pada bulan Juni 2014. Versi yang paling stabil dirilis pada bulan Desember 2014, dimulai dari versi 1.0. Berbasiskan JetBrainns' IntelliJ IDEA, Studio di desain khusus untuk *Android Development*. Android Studio sudah bisa diunduh untuk Windows, Mac OS X, dan Linux

Berikut ini adalah beberapa keunggulan Android Studio:

1. *Android Memory (HPROF) Viewer*

Android Studio kini mengizinkan penggunanya untuk menangkap dan menganalisa *snapshot* memori dalam format Android HPROF.

2. *Allocation Tracker*

Untuk mempermudah dalam menganalisa penggunaan alokasi memori aplikasi yang dibuat oleh penggunanya, *Allocation Tracker* kini menambahkan fitur visual. Dengan adanya fitur ini, alokasi memori yang digunakan oleh aplikasi yang dibuat oleh penggunanya dapat terlihat dalam diagram berbentuk lingkaran.

3. *APK Test*

Kini plugin baru ('com.android.test') telah ditambahkan untuk mempermudah penggunanya dalam mencoba atau melakukan *testing* aplikasi Android yang sedang dikembangkan. Untuk menggunakan fitur ini, penggunanya harus memiliki Gradle Plugin versi 1.3.

4. Application Permission Annotation

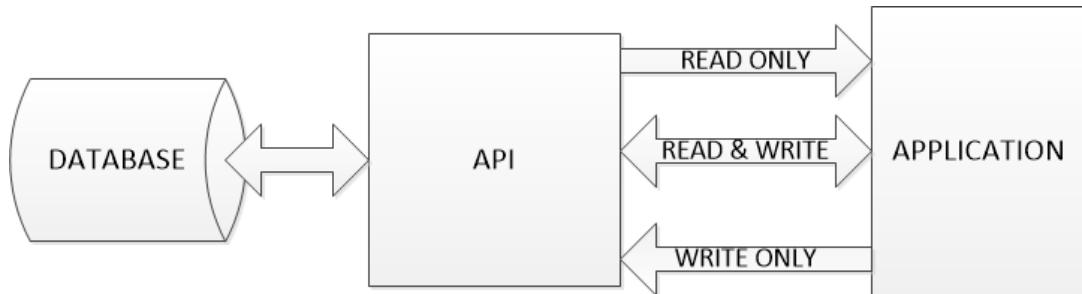
Android Studio (sejak versi 1.3) mendukung *inline code annotation* yang akan membantu penggunanya mengatur *app permission* yang dirilis pada Android M (Android 6.0/Marshmallow).

5. SDK Auto Update dan SDK Manager

Android Studio mengatur pembaharuan SDK dan plugin-plugin lainnya secara otomatis.

2.7 Application Programming Interface (API)

Application Programming Interfaces atau biasa disingkat API, yang didalamnya termasuk kumpulan *library*, *framework*, peralatan penunjang (*toolkits*), dan peralatan pengembangan *software* lainnya, digunakan secara virtual dalam semua bahasa pemrograman [12]. Jika sebuah perangkat lunak mengandung API *internal* (yang berasal dari dalam perangkat lunak tersebut) serta API publik (seperti Java Platform SDK, Windows .NET Framework, jQuery untuk JavaScript, dan Web services seperti Google Map), maka setiap baris dari kode yang ditulis oleh *programmer* memanggil dan menggunakan API. API menyediakan sebuah mekanisme dalam penggunaan kembali sebuah kode yang telah dibuat sebelumnya sehingga *programmer* dapat memanfaatkannya kembali untuk keperluan yang berbeda. Hal ini sangat efektif apabila dibandingkan dengan membuat setiap program dan kode dari awal. Lebih lanjut lagi, penggunaan API sangat dibutuhkan karena akses yang bersifat *low-level* menuju kepada sebuah sumber dari suatu sistem (seperti grafik atau gambar, *networking*, dan *file* sistem) yang tersedia hanya melalui API yang terproteksi.



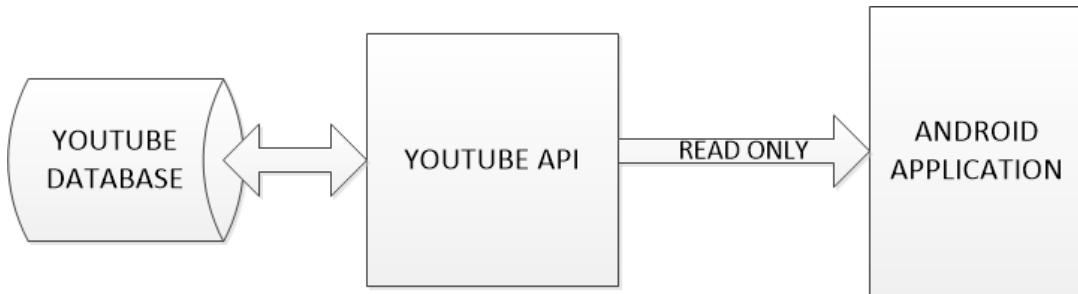
Gambar 2.12: Cara Kerja API

API menjembatani sebuah basis data dengan aplikasi yang mengakses basis data tersebut. Terdapat tiga jenis metode yang dihasilkan oleh API, yaitu metode *read only* (hanya membaca), *write only* (menulis atau mengubah saja), dan *read and write* (membaca dan menulis atau mengubah data). Ketiga jenis metode tersebut dibuat sesuai dengan kebutuhan aplikasi yang akan dijembatani oleh API.

2.8 Integrasi YouTube dengan Aplikasi Android

YouTube dapat diintegrasikan ke dalam sebuah aplikasi Android dengan mengimplementasikan API (*Application Programming Interface*) yang disediakan oleh YouTube dan Google secara gratis (*open source*) yang berarti semua orang dapat menggunakan tanpa terkecuali. API tersebut dinamakan YouTube Android Player API.

Dilansir dari situs Google Developer, API tersebut memungkinkan pengguna untuk memasukkan fungsi pemutaran video ke dalam aplikasi Android. API tersebut mendefinisikan metode untuk menampilkan dan memainkan video-video YouTube (dan juga *playlist*) dan untuk memodifikasi serta mengatur pemutaran video dalam aplikasi Android.



Gambar 2.13: Cara Kerja YouTube API

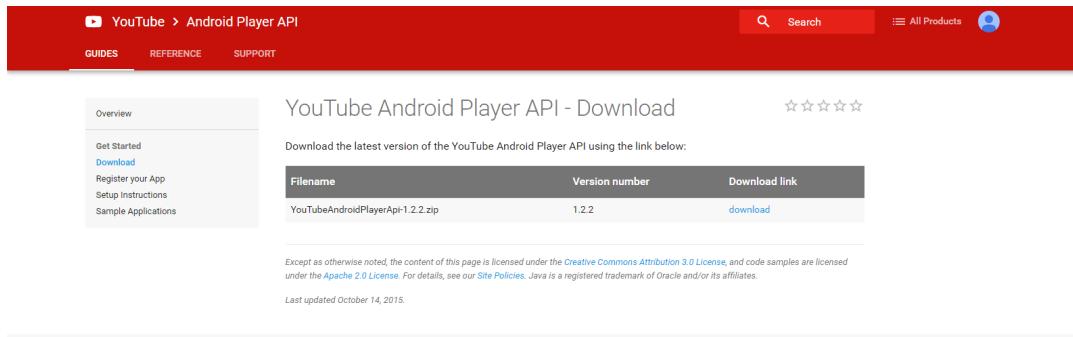
YouTube Android Player API termasuk dalam jenis API yang menyediakan *read only method*, yaitu API yang menyediakan metode untuk membaca data yang terdapat dalam sebuah basis data, dimana basis data yang digunakan pada aplikasi ini adalah basis data YouTube. Video dialirkan kedalam aplikasi Android menggunakan berbagai macam metode yang terdapat dalam API YouTube tersebut.

Dengan menggunakan API tersebut, pengguna dapat menampilkan berbagai video ke dalam sebuah pemutar yang terintegrasi pada *User Interface* (UI) penggunanya. Kemudian penggunanya juga dapat mengatur pemutaran video secara terprogram. Sebagai contoh, pengguna dapat memainkan, melakukan jeda, atau mempercepat video ke titik tertentu pada video yang sedang diputar

Pengguna juga dapat memasukkan *event listener* untuk mendapatkan *callbacks* dari beberapa event, seperti pemuatan video pada pemutar video atau perubahan tahap pada pemutar video. Dan yang terakhir, API tersebut memiliki *helper functionality* (fungsionalitas pembantu) untuk mendukung perubahan orientasi seperti transisi menjadi pemutaran dalam layar penuh (*fullscreen playback*).

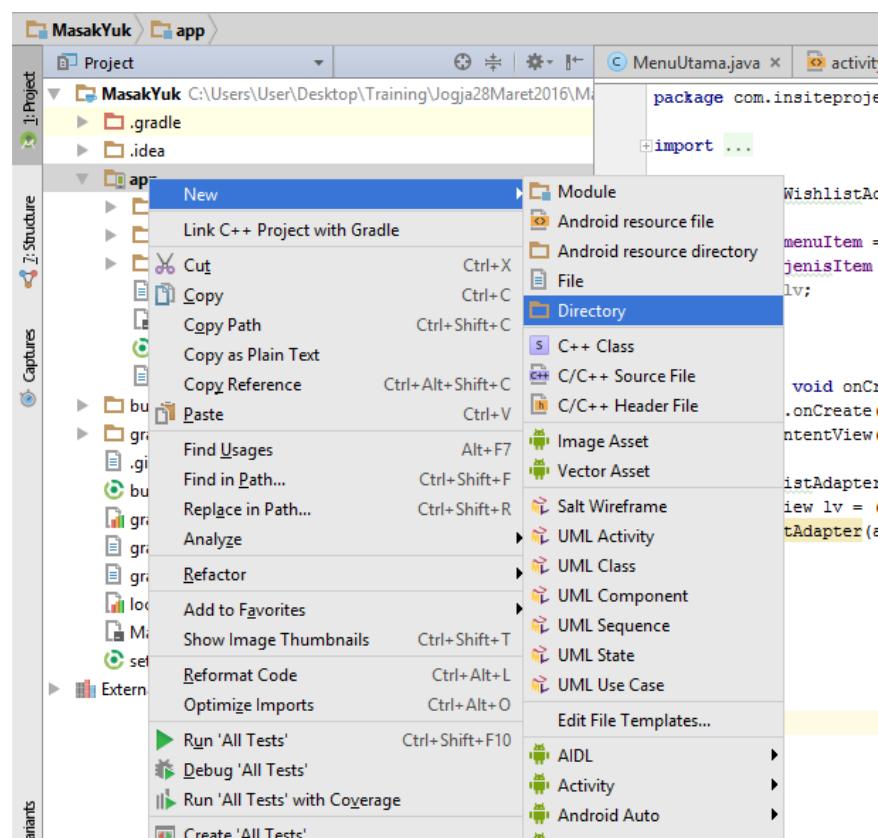
Langkah-langkah untuk menginisialisasi integrasi YouTube dengan aplikasi Android:

1. Unduh terlebih dahulu file YouTube Android Player API pada link <https://developers.google.com/youtube/android/player/downloads/>



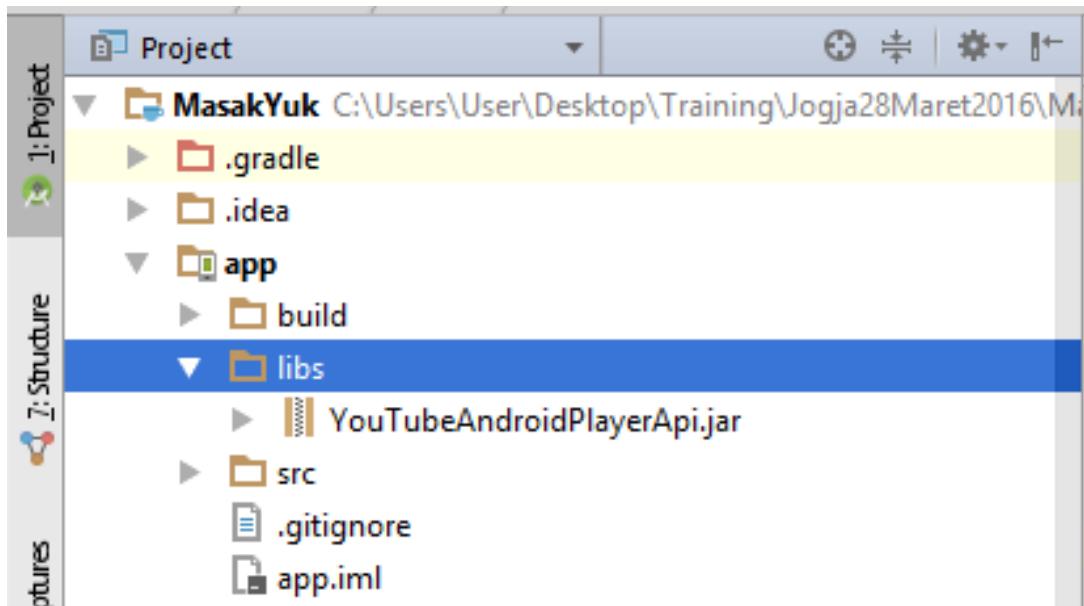
Gambar 2.14: Laman Unduh YouTube Player API

2. Kemudian pada Android Studio, ubah tampilan *tree* dari Android menjadi Project dan buat direktori baru bernama *libs* pada direktori *app*



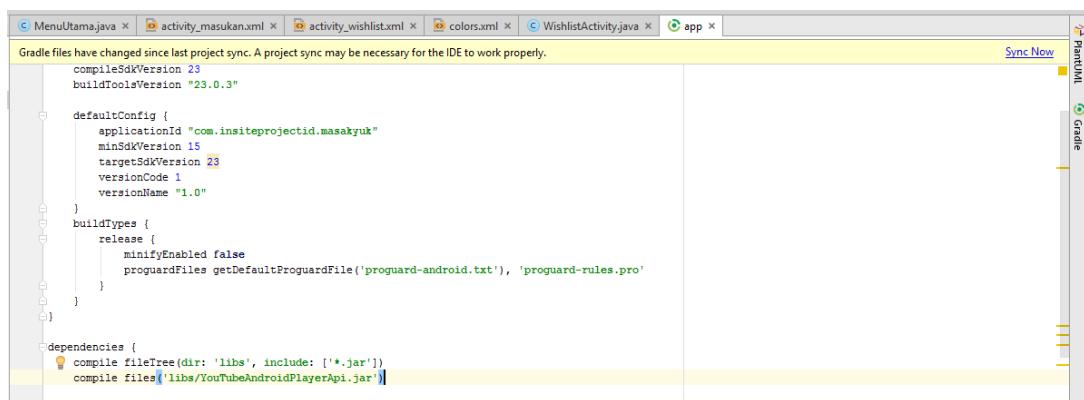
Gambar 2.15: Tampilan *tree Project*

3. Ekstrak berkas yang telah diunduh pada langkah pertama dan salin berkas YouTubeAndroidPlayerApi.jar pada direktori *libs*



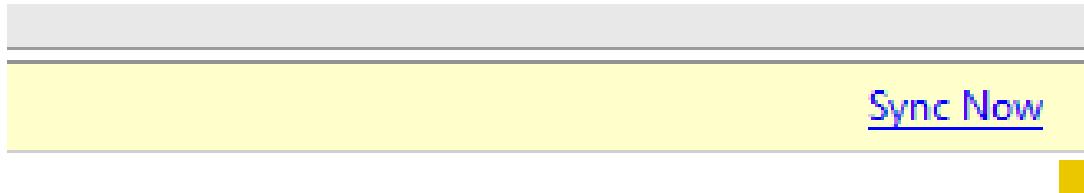
Gambar 2.16: Menyalin Berkas YouTube API

4. Tambahkan kode compile files('libs/YouTubeAndroidPlayerApi.jar') pada berkas build.gradle yang terdapat di dalam direktori *app*



Gambar 2.17: Menambah Baris Kode

5. Akan muncul sebuah pemberitahuan untuk melakukan sinkronisasi pustaka yang ada dengan pustaka YouTube API. Klik *Sync Now* pada pemberitahuan tersebut untuk menginisialisasikan Android dengan YouTube



Gambar 2.18: Sync Now

6. Buat sebuah Activity dan tambahkan baris kode yang sama dengan kode yang tertera di bawah ini

```
public class ResepActivity extends YouTubeBaseActivity implements YouTubePlayer.OnInitializedListener {
```

Gambar 2.19: Menambah Baris Kode pada Activity

7. Inisialisasikan YouTubePlayerView pada Activity tersebut

```
private YouTubePlayerView youtubeView;
```

Gambar 2.20: Inisialisasi YouTubePlayerView

8. Inisialisasikan juga YouTubePlayerView yang sudah disiapkan pada Activity tersebut ke dalam *method* onCreate.

```
youtubeView = (YouTubePlayerView) findViewById(R.id.youtube_view);
youtubeView.initialize(Config.YOUTUBE_API_KEY, this);
```

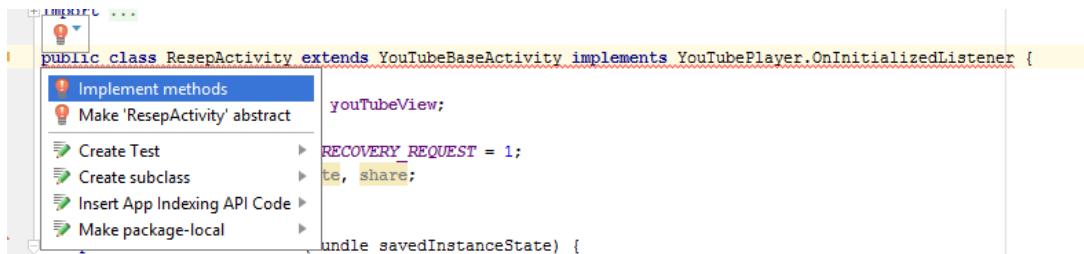
Gambar 2.21: Inisialisasi YouTubePlayerView ke dalam *method* onCreate

9. Pada *layout*, tambahkan baris kode di bawah ini untuk menginisialisasi tampilan YouTube

```
<com.google.android.youtube.player.YouTubePlayerView
    android:id="@+id/youtube_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"/>
```

Gambar 2.22: Inisialisasi Tampilan YouTube

10. Implementasikan *method* yang berasal dari API YouTube dengan melakukan klik pada lampu merah yang terdapat pada inisialisasi *class Activity* seperti yang terdapat di bawah ini



Gambar 2.23: Implementasi *Method* YouTube

11. Akan muncul dua *method* penting dalam mengintegrasikan YouTube ke dalam Android, yaitu onInitializationSuccess dan onInitializationFailure seperti di bawah ini

```
@Override
public void onInitializationSuccess(YouTubePlayer.Provider provider, YouTubePlayer youTubePlayer, boolean b) {
}

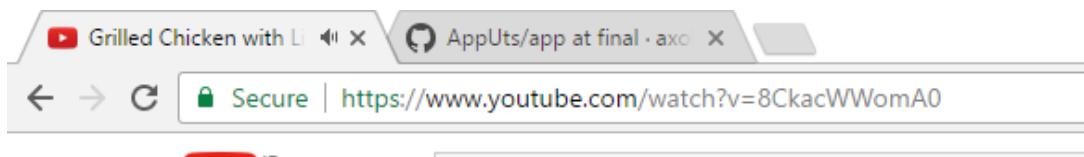
@Override
public void onInitializationFailure(YouTubePlayer.Provider provider, YouTubeInitializationResult errorReason) {
}
```

Gambar 2.24: Inisialisasi Tampilan YouTube

onInitializationSuccess adalah method yang berfungsi untuk menjalankan fitur-fitur YouTube di dalam aplikasi Android apabila inisialisasi sukses. Sedangkan

onInitializationFailure adalah metode untuk mengambil alih tindakan yang terjadi apabila YouTube gagal diinisialisasikan.

12. Untuk memutar sebuah video YouTube, API membutuhkan masukan berupa karakter alfanumerik yang terdapat pada akhir dari link YouTube.



Gambar 2.25: Contoh Karakter Alfanumerik YouTube

Berikut adalah contoh pengaplikasiannya

```
@Override
public void onInitializationSuccess(YouTubePlayer.Provider provider, YouTubePlayer youtubePlayer, boolean b) {
    if (!b) {
        youtubePlayer.cueVideo("8CkacWWomA0");
    }
}
```

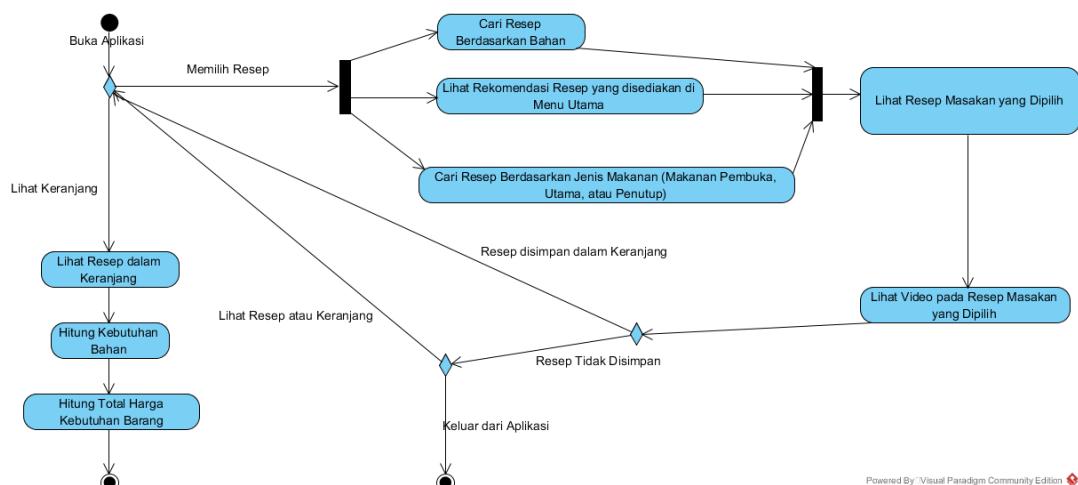
Gambar 2.26: Contoh Pengaplikasian Karakter Alfanumerik YouTube

API *client library* berinteraksi dengan sebuah layanan yang terdistribusi sebagai sebuah bagian dari aplikasi YouTube untuk platform Android. *Client library* tersebut meninggalkan sedikit jejak kaki, dengan kata lain tidak akan membebani atau menjadikan ukuran file aplikasi menjadi lebih besar

BAB III

IMPLEMENTASI PROGRAM

Sesuai dengan tahapan-tahapan pengembangan perangkat lunak yang tertera pada SDLC, maka penulis melakukan serangkaian kegiatan untuk menunjang pengembangan aplikasi resep masakan berbasis Android ini. Karena menggunakan SDLC Model Spiral, maka penulis akan melakukan beberapa tahapan yaitu identifikasi, desain, konstruksi dan pembangunan, serta evaluasi. Evaluasi akan dibahas pada Bab IV (Uji Coba dan Hasil Percobaan) Adapun alur dari aplikasi yang akan dibuat dijelaskan pada Gambar 3.1



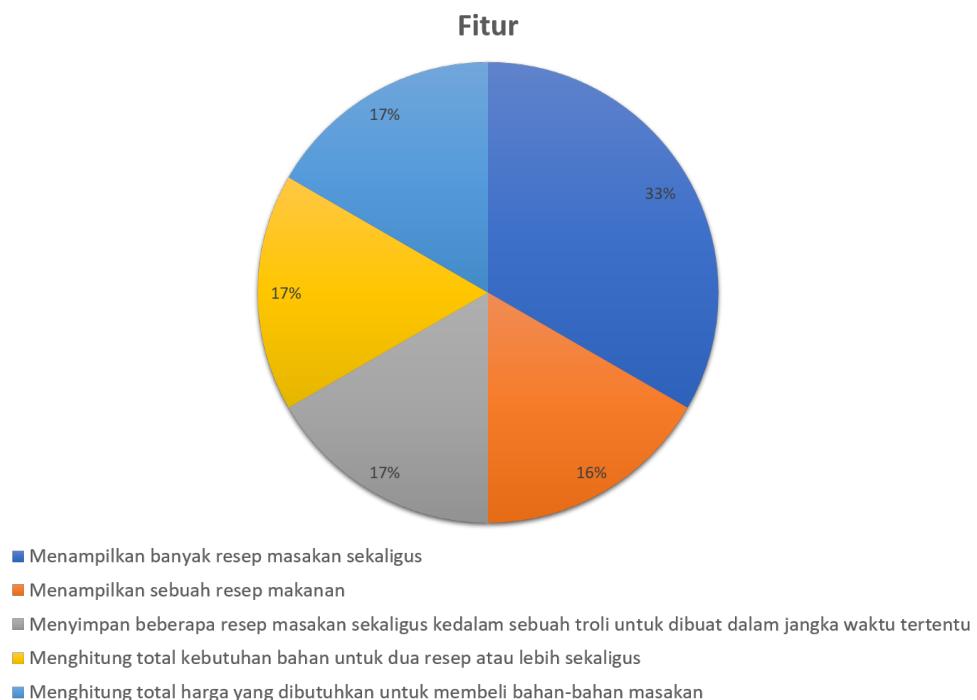
Gambar 3.1: Alur Kerja Aplikasi

3.1 Identifikasi

Penulis mengumpulkan beberapa informasi yang dibutuhkan untuk membangun aplikasi ini dengan cara menyebarkan beberapa kuisioner kepada ahli di bidang masakan, baik itu chef, asisten chef, serta pemilik usaha dibidang kuliner untuk mengetahui fitur apa sajakah yang seharusnya ada dalam aplikasi berbasis Android ini.

Penulis juga tidak lupa meminta pandangan kepada beberapa kaum awam yang berasal dari berbagai kalangan untuk mengetahui tentang pendapat masyarakat tentang fitur apa saja yang seharusnya terdapat pada aplikasi resep masakan ini.

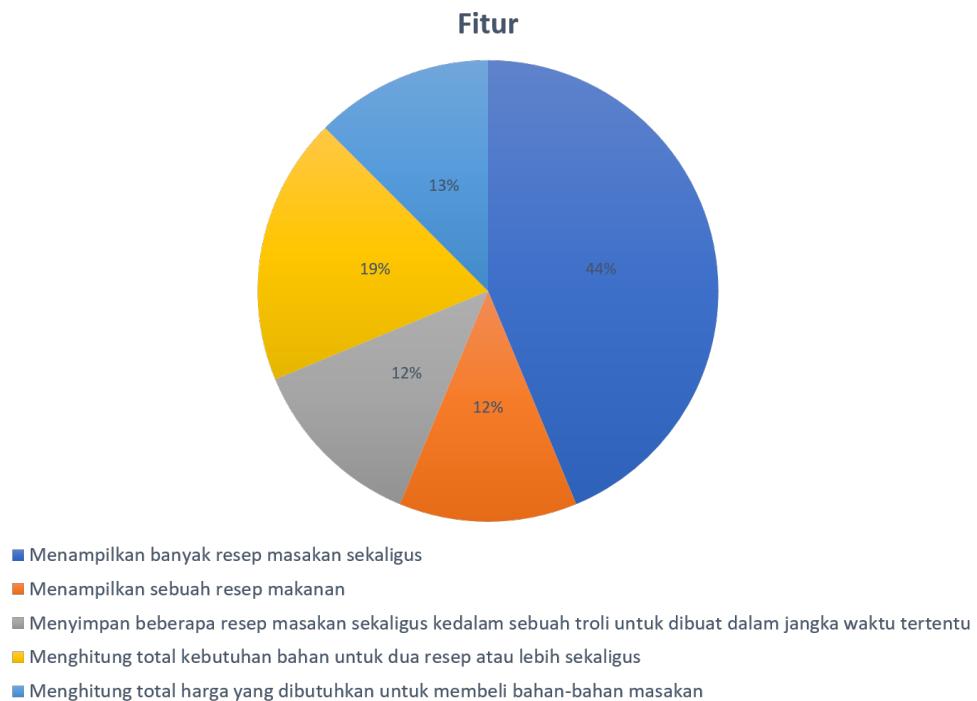
Dari 4 responden yang merupakan para ahli, dimana 2 dari 4 responden tersebut adalah chef, satu orang asisten chef, dan seorang pemilik usaha kuliner, fitur menampilkan banyak resep masakan sekaligus merupakan fitur yang paling banyak dipilih dengan nilai 2. Sedangkan fitur lain yang diusulkan penulis seperti menampilkan sebuah resep makanan, menyimpan beberapa resep masakan sekaligus kedalam sebuah troli untuk dibuat dalam jangka waktu tertentu, menghitung total kebutuhan bahan untuk dua resep atau lebih sekaligus, serta menghitung total harga yang dibutuhkan untuk membeli bahan-bahan masakan mendapatkan masing-masing nilai 1.



Gambar 3.2: Pie Chart Hasil Kuisioner Ahli

Untuk responden yang berasal dari kaum awam yang berjumlah 9 orang, dimana 2 dari 9 orang responden berprofesi sebagai guru, satu orang berprofesi sebagai

ibu rumah tangga, sedangkan sisanya adalah seorang karyawan swasta, fitur menampilkan banyak resep masakan sekaligus merupakan fitur yang paling banyak dipilih dengan nilai 7. Sementara fitur menghitung total kebutuhan bahan untuk dua resep atau lebih sekaligus berada pada peringkat 2 dengan nilai 3. Sedangkan fitur lain yang diusulkan penulis seperti menampilkan sebuah resep makanan, menyimpan beberapa resep masakan sekaligus kedalam sebuah troli untuk dibuat dalam jangka waktu tertentu, serta menghitung total harga yang dibutuhkan untuk membeli bahan-bahan masakan mendapatkan masing-masing nilai 2.



Gambar 3.3: *Pie Chart* Hasil Kuisioner Awam

Masyarakat awam yang menjadi responden pada kuisioner yang telah disebarkan oleh Penulis menyertakan beberapa pendapat maupun ide tentang fitur apa yang seharusnya dimiliki oleh sebuah sistem aplikasi resep masakan berbasis Android, diantaranya adalah:

- Memberikan deskripsi bahan-bahan makanan yang dibutuhkan juga dan bisa

didapatkan dimana

- Dalam sebuah resep akan lebih baik jika ada detail resep ini untuk berapa porsi, sehingga orang awam dapat mengetahui berapa takaran bahan-bahan masakan yang akan digunakan untuk porsi tertentu
- Menampilkan video tutorial masak, dan alternatif bahan masakan (bahan masakan pengganti untuk alergi misalnya)
- Aplikasi disertai video pembuatan masakan dan resepnya.

Sedangkan dari usulan maupun ide yang datang dari para professional atau ahli di bidang ini adalah sebagai berikut:

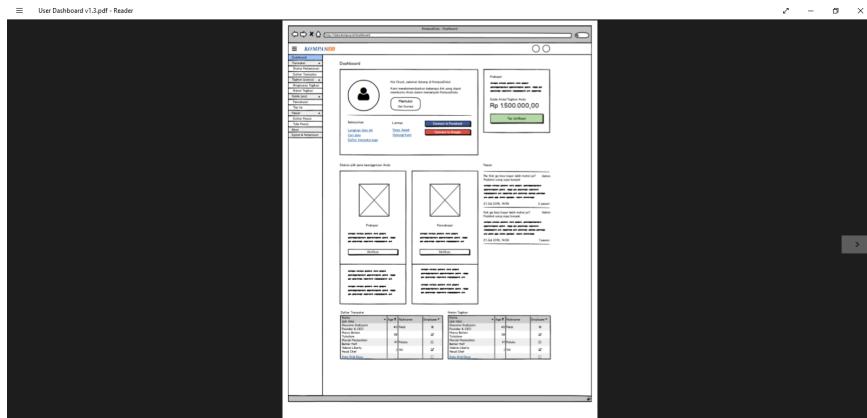
- Memberikan pilihan apa saja yang dapat dimasak dari suatu bahan-bahan tertentu
- Mengetahui kapan bahan akan habis
- Menampilkan metode memasak yang jelas sesuai dengan standar makanan yang akan dibuat
- Informasi jumlah takaran dalam sebuah resep haruslah jelas
- Mungkin bisa ditambah dengan foto atau video pembuatannya
- Memberikan tips dan trik
- Memberikan keterangan pada tulisan berbahasa asing untuk menambah pengetahuan. Contohnya *Mise En Place* artinya preparasi

Melalui data yang telah terhimpun, penulis memutuskan untuk memilih fitur yang banyak dipilih oleh responden, baik ahli maupun kaum awam, untuk diimplementasikan pada aplikasi yang akan dibuat oleh penulis, yaitu menampilkan banyak

resep masakan sekaligus, menghitung total kebutuhan bahan untuk dua resep atau lebih sekaligus, menyimpan beberapa resep masakan sekaligus kedalam sebuah troli untuk dibuat dalam jangka waktu tertentu, serta menghitung total harga yang dibutuhkan untuk membeli bahan-bahan masakan. Sementara, fitur tambahan yang diinisiasi oleh penulis adalah fitur mencari resep berdasarkan bahan masakan yang akan dibuat.

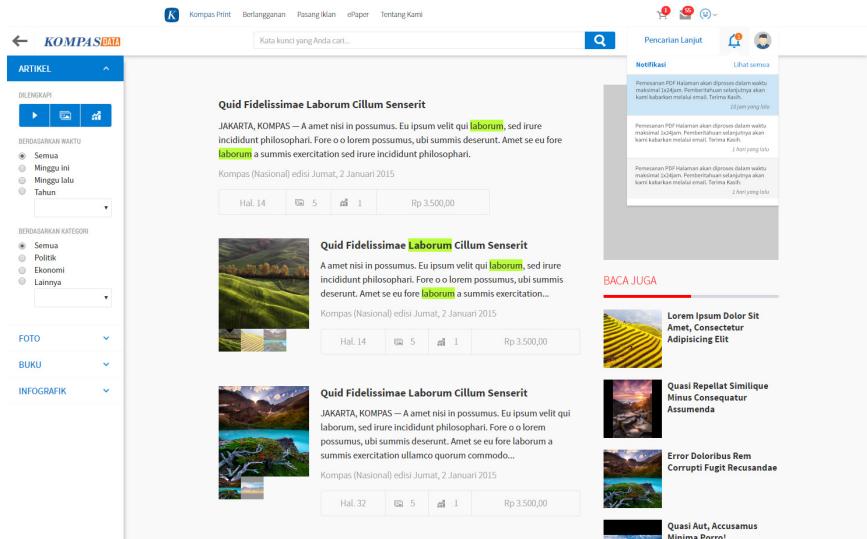
3.2 Desain

Dalam mengembangkan aplikasi ini, penulis mengembangkan desain aplikasi terlebih dahulu, mulai dari desain *Use Case*, *Entity Relationship Diagram* (ERD), *Class Diagram*, serta *Activity Diagram*. Selain itu, penulis juga membuat desain *wireframe* dan *mock-up*. Penulis memutuskan untuk membuat kedua desain tersebut untuk memudahkan pengguna dalam mengeksekusi desain antarmuka dari aplikasi ini. Berdasarkan pengalaman penulis dalam mengikuti program Praktek Kerja Lapangan pada PT Kompas Media Nusantara (Harian Kompas), dimana penulis berperan sebagai *Front-End Developer*, perbedaan dari *wireframe* dan *mock-up* sendiri adalah pada tampilannya. Wireframe mengedepankan *User Experience* dari sebuah laman aplikasi, yaitu alur kerja dari laman aplikasi tersebut. Biasanya tampilan yang disuguhkan oleh *wireframe* tidak terlalu menarik dan hanya terlihat seperti kerangka dari tampilan aplikasi tersebut.



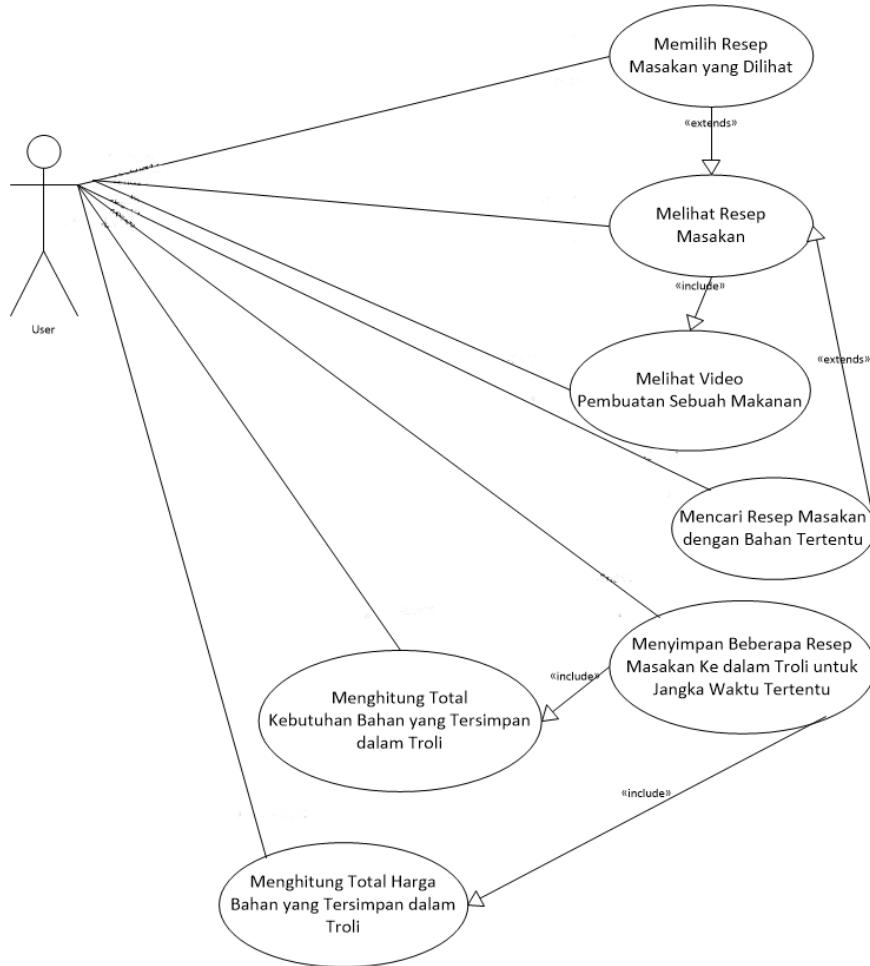
Gambar 3.4: Contoh Wireframe

Sedangkan *mock-up* adalah sebuah rancangan laman antarmuka aplikasi yang bersifat dekoratif dengan dekorasi yang sudah didefinisikan secara jelas oleh pembuatnya. *Mock-up* menonjolkan sisi *User Interface* dari sebuah laman antarmuka aplikasi. Berbeda dengan *wireframe* yang hanya mengedepankan alur kerja dari sebuah laman, *mock-up* adalah pengembangan lebih lanjut dari *wireframe* sehingga sisi estetika dari sebuah rancangan antarmuka aplikasi lebih menonjol. *Mock-up* memudahkan seorang Front-End Developer untuk mengembangkan dan mengeksekusi sebuah laman antarmuka aplikasi.



Gambar 3.5: Contoh Mock-Up

3.2.1 Use Case Diagram

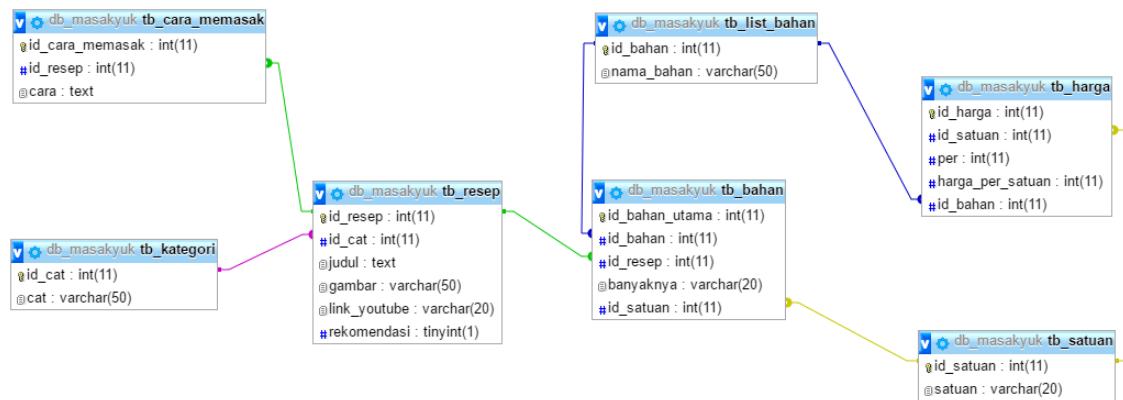


Use Case Diagram tersebut menunjukkan bagaimana pengguna aplikasi (*user*) berinteraksi dengan aplikasi yang dikembangkan oleh peneliti. Dalam *use case* tersebut dijelaskan bahwa *user* dapat memilih resep masakan yang ingin dilihat pada menu utama untuk kemudian melihat resep masakan sekaligus melihat video cara membuat makanan yang terdapat dalam resep tersebut. Selain itu, *user* juga dapat mencari resep masakan berdasarkan bahan tertentu sesuai dengan keinginan *user*. Setelah dilihat, *user* dapat menyimpan resep masakan tersebut ke dalam Troli, dimana pada aplikasi ini disebut sebagai *Wishlist*. *User* dapat menyimpan sampai 3 resep masakan sekaligus dalam *Wishlist*. Resep yang ada di dalam *wishlist* dapat dihitung total ke-

butuhan bahannya dan juga dapat menghitung total kebutuhan harga untuk membeli bahan-bahan tersebut.

3.2.2 Entity Relationship Diagram

ERD pada aplikasi ini berfungsi untuk menyimpan resep, kategori resep, bahan-bahan yang ada pada resep, cara memasak, serta harga pada tiap-tiap bahan yang tercantum dalam resep. Untuk itu, penulis membuat beberapa tabel untuk menunjang penyimpanan data dari tiap elemen tersebut.



Gambar 3.6: Desain ERD Aplikasi yang Dikembangkan Penulis

Penulis menyediakan tabel resep untuk menyimpan data-data yang berhubungan dengan resep. Kemudian, penulis juga menyediakan tabel bahan untuk menyimpan bahan-bahan yang terdapat pada sebuah resep serta tabel list atau daftar bahan untuk menyimpan data bahan-bahan secara umum yang dijadikan sebagai referensi dari tabel bahan. Penulis juga menyediakan tabel harga dan satuan untuk melengkapi tabel bahan. Tabel satuan dibuat karena adanya kemungkinan sebuah bahan dapat memiliki banyak jenis satuan, sedangkan tabel harga dibuat untuk menghitung harga bahan yang tertera dalam tabel bahan dengan kalkulasi yang berbeda bergantung pada satuannya. Selain itu, penulis juga membuat tabel cara memasak untuk

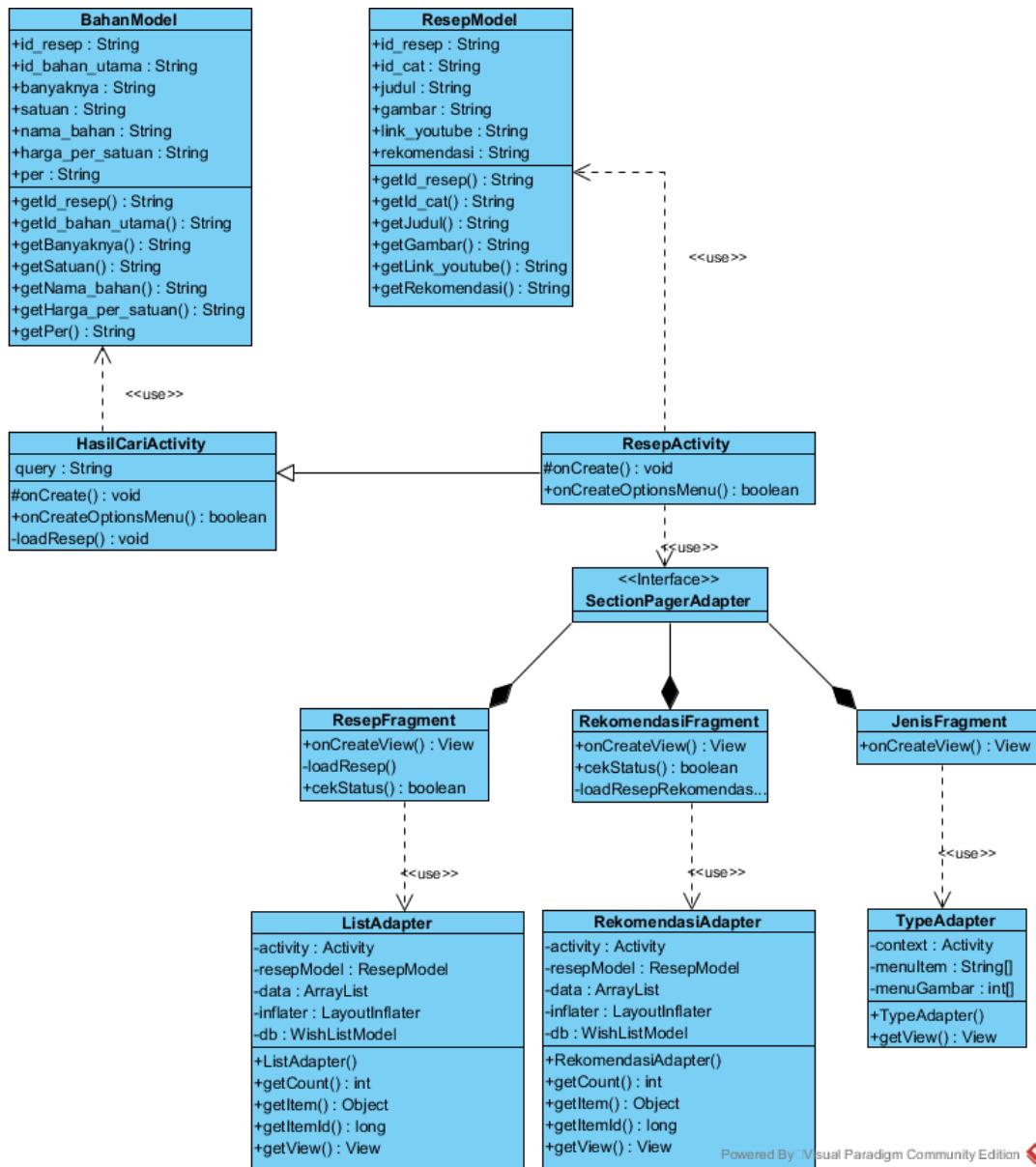
menyimpan cara-cara memasak berdasarkan resep yang ada. Terakhir, penulis membuat tabel kategori untuk mengelompokkan resep berdasarkan jenisnya yaitu Makanan Pembuka, Makanan Utama, dan Makanan Penutup.

Tabel resep berelasi *one-to-many* dengan tabel cara memasak dan tabel bahan karena satu resep dapat memiliki banyak cara memasak dan banyak bahan. Sedangkan tabel kategori berelasi *one-to-many* juga karena satu kategori dapat memiliki banyak resep. Tabel list bahan berelasi *one-to-many* dengan tabel bahan karena bahan yang terdapat pada tabel list bahan dapat dimasukkan lebih dari satu kali pada tabel bahan yang secara langsung menunjang tabel resep. Tabel Satuan berelasi one-to-many dengan tabel harga dan tabel bahan karena tabel harga dapat memiliki banyak satuan serta tabel bahan yang juga sama, yakni dapat memiliki lebih dari satu satuan.

3.2.3 *Class Diagram*

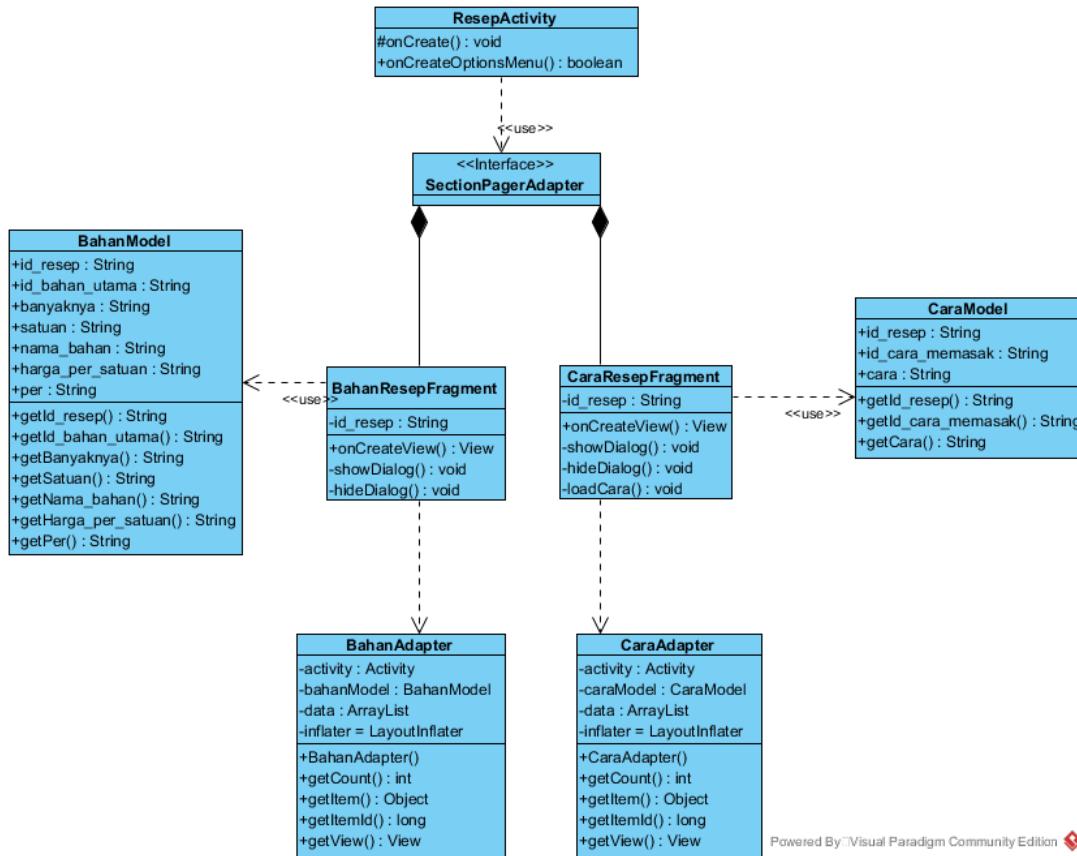
Pada pengembangan aplikasi resep masakan ini, penulis memecah *class diagram* menjadi tiga bagian, yaitu *class diagram* menu utama, *class diagram* detail resep, dan *class diagram* wishlist.

Terdapat dua model yang diakses pada *class diagram* Menu Utama. Resep Model digunakan untuk menyediakan data resep yang diolah pada *controller* Resep Fragment, Rekomendasi Fragment, dan Jenis Fragment. Bahan Model digunakan untuk menyediakan data resep berdasarkan bahan yang dicari oleh *user*. *Controller* selanjutnya mengalirkan data yang telah diolah kepada *view* yang terdapat pada masing-masing *controller*. Desain class diagram Mennu Utama dapat dilihat pada Gambar 3.7.



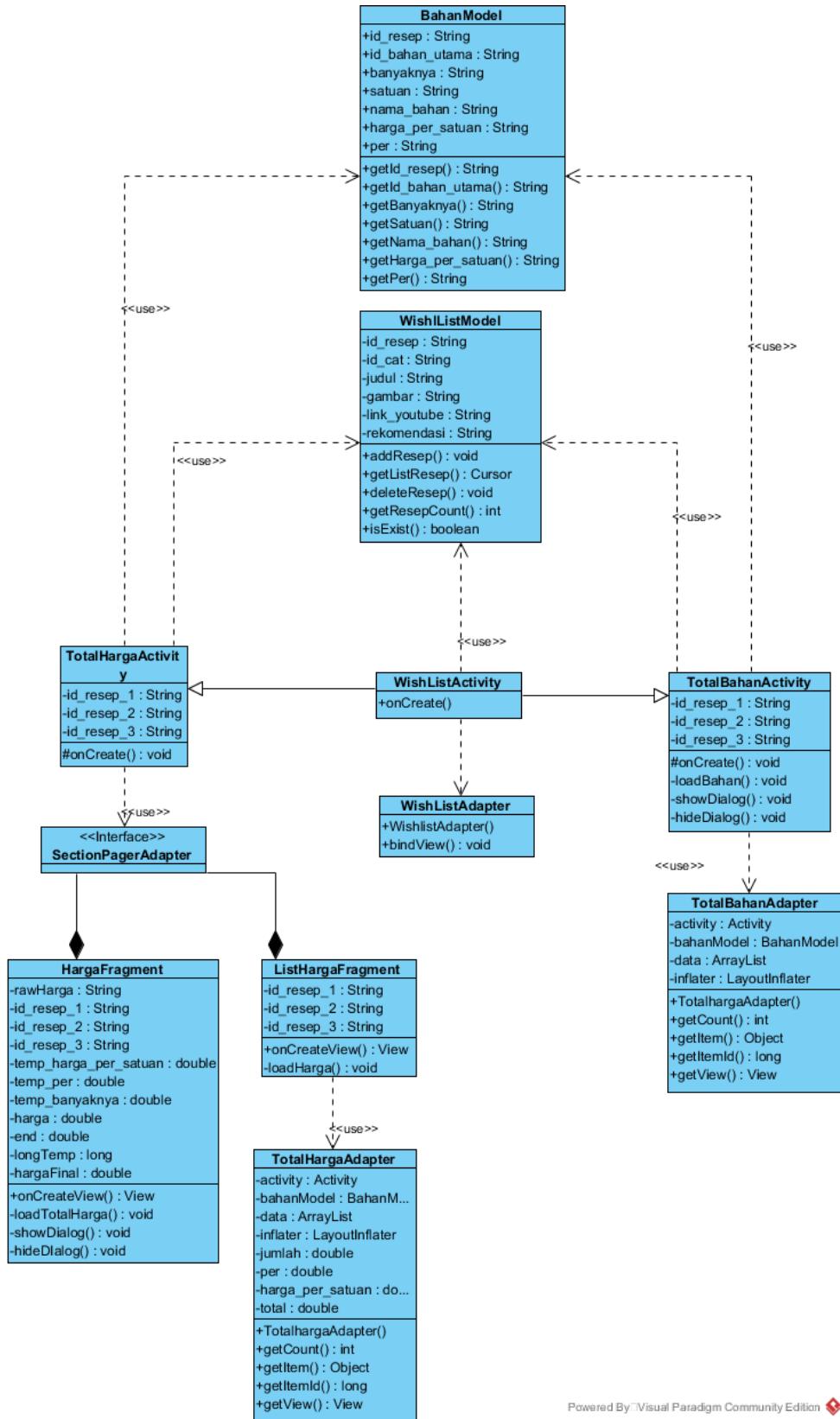
Gambar 3.7: Class Diagram Menu Utama

Class diagram Detail Resep, yang terdapat pada Gambar 3.8, memerlukan dua buah model yaitu Bahan Model dan Cara Model. Kedua model tersebut diperlukan untuk menyediakan data bahan yang dibutuhkan serta cara memasak dalam sebuah resep masakan. Data tersebut kemudian diolah dalam *controller* BahanResepFragment dan CaraResepFragment untuk kemudian dialirkan kepada masing-masing *view*.



Gambar 3.8: Class Diagram Detail Resep

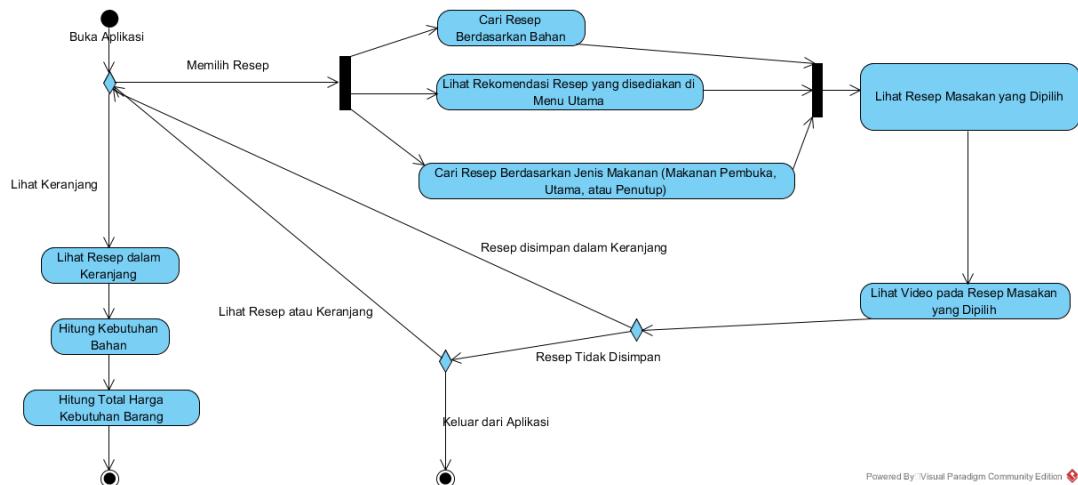
Wishlist merupakan *class diagram* terakhir yang dibuat oleh penulis. Terdapat dua model yang digunakan dalam *class diagram* ini yakni Bahan Model dan WishList Model. WishList Model digunakan untuk mengalirkan data resep yang telah disimpan oleh pengguna sebelumnya untuk kemudian dilakukan pencocokan data dengan basis data bahan (Bahan Model). Setelah dicocokkan, pengguna atau *user* dapat melakukan perhitungan total bahan yang dibutuhkan serta melakukan perhitungan total harga bahan yang dibutuhkan oleh pengguna beserta dengan daftar harga tiap bahan yang dibutuhkan. Gambar *class diagram* wishlist dapat dilihat pada Gambar 3.9.



Gambar 3.9: Class Diagram Wishlist

3.2.4 Activity Diagram

Alur kerja aplikasi yang dikembangkan oleh penulis atau peneliti dapat dilakukan melalui *activity diagram* yang terdapat pada Gambar 3.10



Gambar 3.10: Activity Diagram dari Aplikasi yang Dibuat Penulis

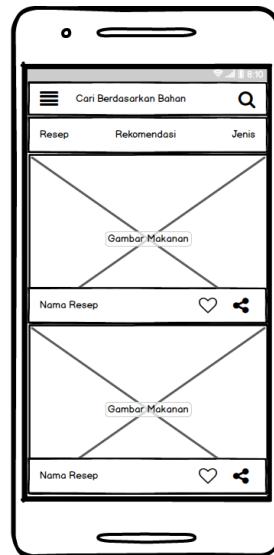
Pada awal membuka aplikasi, pengguna aplikasi dapat memilih resep atau ingin melihat keranjang atau *wishlist* apabila sudah pernah memasukkan resep ke dalam keranjang sebelumnya. Apabila pengguna ingin melihat resep, maka pengguna dapat memilih resep dengan cara melihat daftar resep, mencari resep berdasarkan bahan, melihat rekomendasi resep yang disediakan, serta mencari resep berdasarkan jenis makanan (makanan pembuka, makanan utama, dan makanan penutup). Setelah itu, pengguna melihat resep yang telah dipilih serta dapat melihat video pembuatannya juga. Apabila resep disimpan kedalam keranjang, maka pengguna dapat kembali memilih resep atau melihat keranjang langsung untuk menghitung kebutuhan bahan serta total harga kebutuhan bahan. Sedangkan apabila resep masakan yang telah dilihat tidak dimasukkan ke dalam keranjang, maka pengguna dapat memilih resep masakan kembali, melihat keranjang yang sudah diisi sebelumnya, atau dapat keluar dari aplikasi.]

3.2.5 Desain *Wireframe* atau Kerangka Desain



Gambar 3.11: Halaman Awal

Desain halaman selamat datang ketika aplikasi pertama kali dimulai ditunjukkan pada Gambar 3.11. Terdapat logo serta gambar latar belakang yang dibuat oleh penulis. Setelah tiga detik, pengguna akan diarahkan kepada laman utama aplikasi yang terdapat pada Gambar 3.12



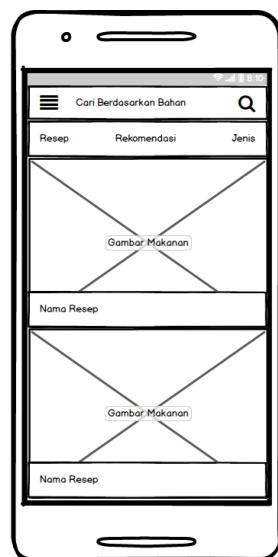
Gambar 3.12: Menu Utama

Terdapat banyak resep masakan yang dapat dipilih pada laman ini. Pada laman menu utama, apabila pengguna mengusap layar dari rah kiri ke kanan, maka akan muncul menu navigasi seperti pada Gambar 3.13. Pada menu navigasi, pengguna dapat memilih untuk melihat *wishlist*, memberi masukan unuk aplikasi, membagikan aplikasi ke media sosial, dan keluar dari aplikasi



Gambar 3.13: Navigasi

Kembali lagi pada menu utama, apabila pengguna mengusap layar ke kanan atau menekan tab rekomendasi, maka akan muncul resep masakan yang direkomendasikan oleh aplikasi.



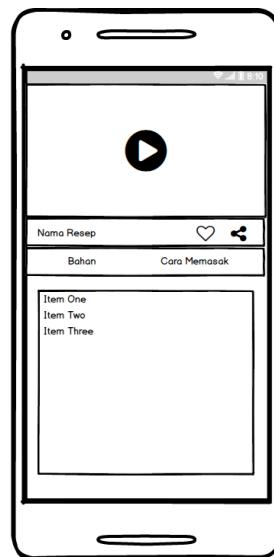
Gambar 3.14: Menu Utama Tab Rekomendasi

Pada tab rekomendasi, apabila pengguna mengusap layar ke arah kanan, maka pengguna akan menemukan pilihan penampilan resep berdasarkan jenis makanannya yaitu makanan pembuka, makanan utama, dan makanan penutup.



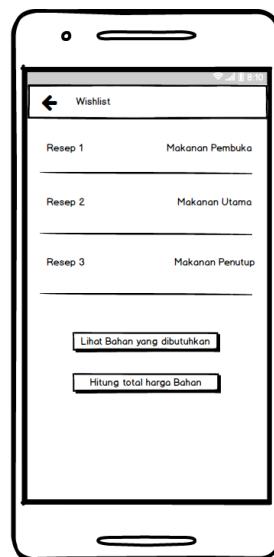
Gambar 3.15: Menu Utama *Tab Jenis*

Apabila pengguna memilih satu dari resep yang berada pada list yang disediakan dari menu-menu yang telah disebutkan sebelumnya, maka akan muncul laman detail resep yang terdiri dari video, bilah bahan resep serta bilah cara memasak dari resep tersebut



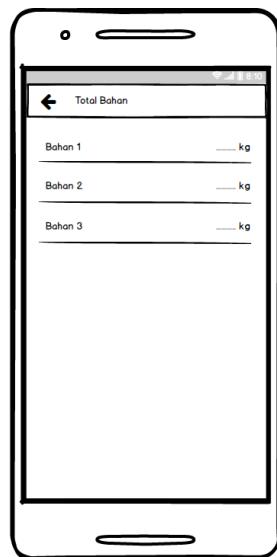
Gambar 3.16: Detail Resep

Pengguna juga dapat melihat *wishlist* dengan memanggil menu navigasi seperti pada Gambar 3.13 dan memilih menu *wishlist*. Laman *wishlist* akan muncul seperti pada Gambar 3.17



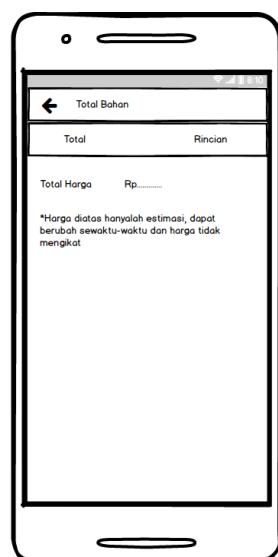
Gambar 3.17: Keranjang atau *Wishlist*

Pada laman *wishlist*, pengguna dapat menghitung total bahan yang dibutuhkan untuk memasak berdasarkan resep yang telah dipilih dan ada pada keranjang



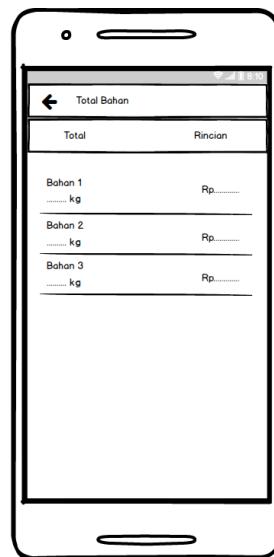
Gambar 3.18: Total Bahan

Selain itu, pada laman *wishlist*, pengguna juga dapat menghitung total harga dari bahan-bahan dari resep yang telah masuk dalam *wishlist*.



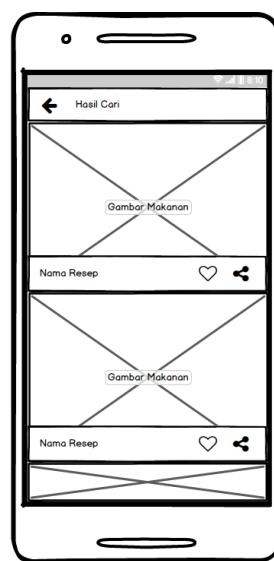
Gambar 3.19: Total Harga

Pengguna juga dapat melihat rincian harga dari tiap bahan yang dibutuhkan oleh resep-resep yang terdapat pada *wishlist*.



Gambar 3.20: Rincian Total Harga

Kembali ke menu utama, pengguna juga dapat melakukan pencarian resep berdasarkan bahan yang dimiliki oleh pengguna.



Gambar 3.21: Hasil Pencarian

Dan terakhir, desain *wireframe* dari laman beri masukan yang dapat diakses melalui menu navigasi.



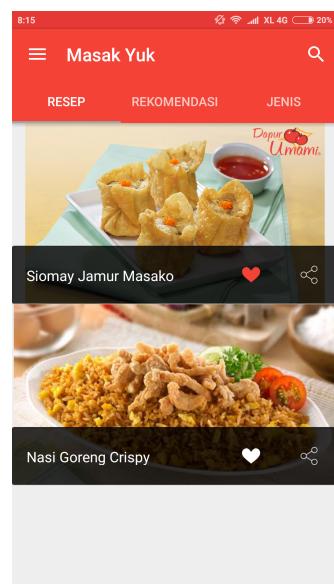
Gambar 3.22: Halaman Beri Masukan

3.2.6 Desain *Mock-Up* atau *User Interface*

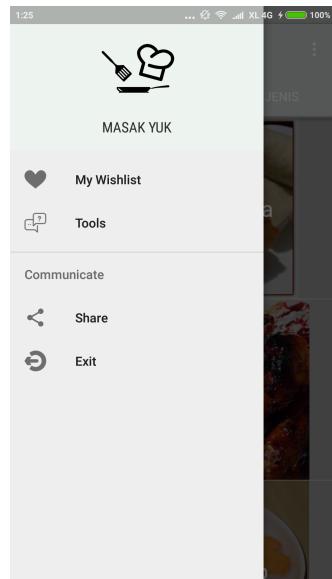
Setelah membuat wireframe, penulis kemudian langsung mengimplementasikan *wireframe* tersebut ke dalam bentuk *mock-up*. Dalam pengembangannya, *mock-up* langsung dibuat dengan menggunakan xml sehingga mempercepat penulis dalam mengembangkan aplikasi resep masakan ini karena desain dari aplikasi tersebut sudah dikembangkan sekaligus dengan *mock-up*. Selanjutnya penulis hanya perlu mengimplementasikan sistem *Back-End* yang bekerja pada aplikasi resep masakan ini. Pengembangan *mock-up* dapat dilihat pada Gambar 3.23 sampai Gambar 3.35. Urutan pengembangan *mock-up* sesuai dengan urutan pengembangan *wireframe*.



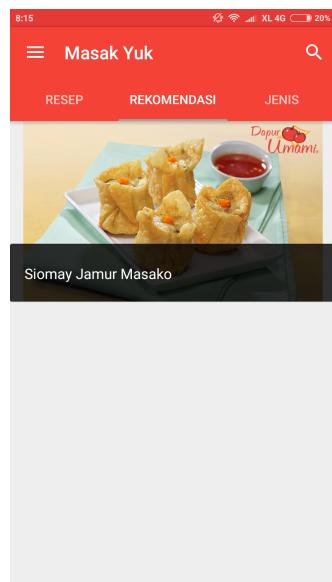
Gambar 3.23: Halaman Awal atau Halaman Selamat Datang



Gambar 3.24: Halaman Utama



Gambar 3.25: Menu Navigasi



Gambar 3.26: Halaman Utama Tab Rekomendasi



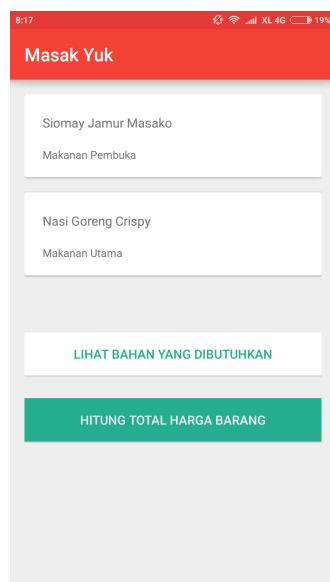
Gambar 3.27: Halaman Utama Tab Jenis



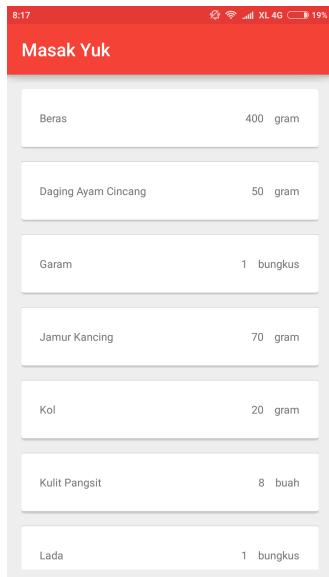
Gambar 3.28: Detail Bahan Resep



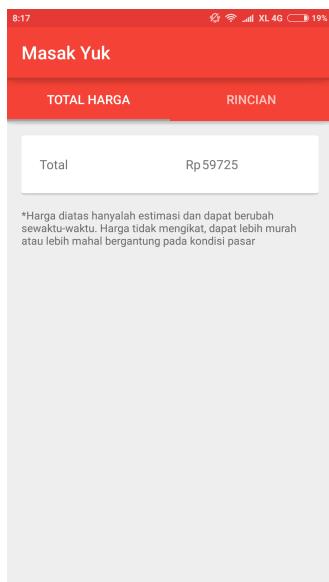
Gambar 3.29: Detail Cara Memasak



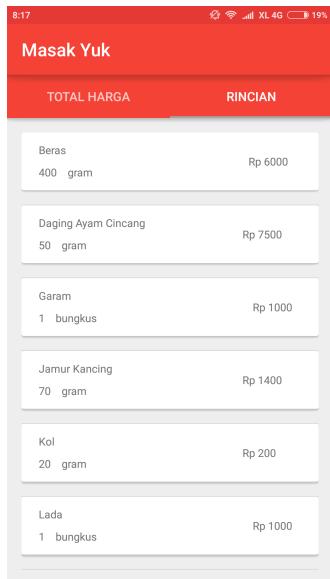
Gambar 3.30: Wishlist



Gambar 3.31: Total Bahan

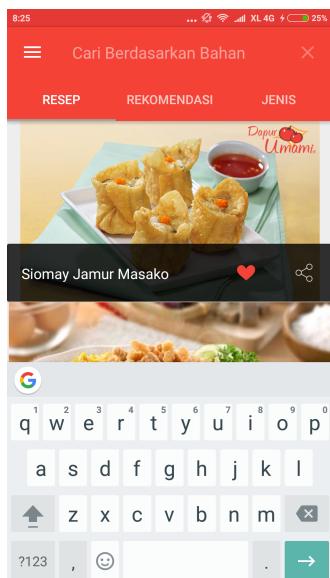


Gambar 3.32: Total Harga

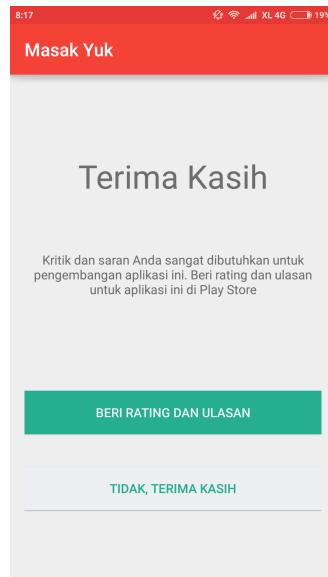


Masak Yuk		
TOTAL HARGA	RINCIAN	
Beras 400 gram	Rp 6000	
Daging Ayam Cincang 50 gram	Rp 7500	
Garam 1 bungkus	Rp 1000	
Jamur Kancing 70 gram	Rp 1400	
Kol 20 gram	Rp 200	
Lada 1 bungkus	Rp 1000	

Gambar 3.33: Rincian Total Harga



Gambar 3.34: Tab Pencarian



Gambar 3.35: Laman Beri Masukan atau *Feedback*

3.3 Konstruksi dan Pembangunan

Tahapan-tahapan dalam proses konstruksi dan pembangunan aplikasi berbasis resep masakan ini adalah sebagai berikut:

1. Membuat Basis Data
2. Membuat API untuk Mengakses Basis Data
3. Melakukan Input Data ke Basis Data
4. Implementasi Desain Aplikasi
5. Implementasi Metode Pertukaran Data antara Aplikasi dengan Basis Data
6. Mengintegrasikan YouTube dengan Aplikasi Android
7. Mengimplementasi Fitur Lainnya

3.3.1 Membuat Basis Data

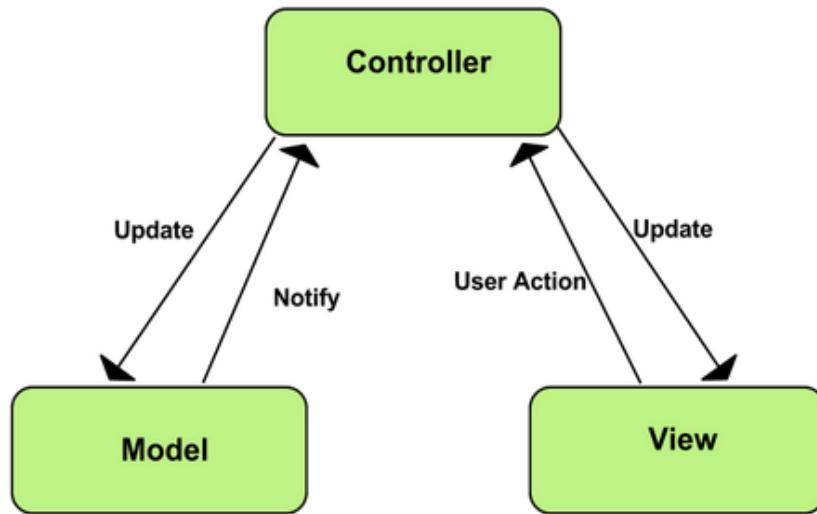
Basis data dibuat dengan menggunakan Basis Data MySQL atau MariaDB. Untuk mempercepat proses penggerjaan, penulis menggunakan aplikasi phpMyAdmin dalam membuat basis data secara keseluruhan, termasuk tabel dan entitas-entitas serta relasi yang terdapat di dalamnya. Basis data dibuat sesuai dengan ERD yang telah dibuat pada tahap desain.

Table	Action	Rows	Type	Collation	Size	Overhead
tb_bahan	Browse Structure Search Insert Empty Drop	18	InnoDB	latin1_swedish_ci	80 Kib	-
tb_cara_memasak	Browse Structure Search Insert Empty Drop	10	InnoDB	latin1_swedish_ci	32 Kib	-
tb_harga	Browse Structure Search Insert Empty Drop	17	InnoDB	latin1_swedish_ci	48 Kib	-
tb_kategori	Browse Structure Search Insert Empty Drop	3	InnoDB	latin1_swedish_ci	16 Kib	-
tb_list_bahan	Browse Structure Search Insert Empty Drop	17	InnoDB	latin1_swedish_ci	16 Kib	-
tb_resep	Browse Structure Search Insert Empty Drop	2	InnoDB	latin1_swedish_ci	32 Kib	-
tb_satuan	Browse Structure Search Insert Empty Drop	9	InnoDB	latin1_swedish_ci	16 Kib	-
tb_user	Browse Structure Search Insert Empty Drop	1	MyISAM	latin1_swedish_ci	6.6 Kib	3,560 B
8 tables	Sum	77	InnoDB	latin1_swedish_ci	246.6 Kib	3.5 Kib

Gambar 3.36: Basis Data yang Dibuat dengan phpMyAdmin

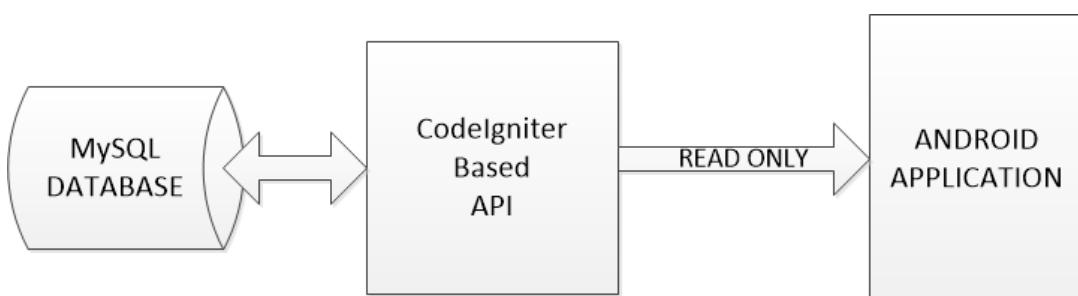
3.3.2 Membuat API untuk Mengakses Basis Data

Untuk dapat mengakses data yang terdapat pada basis data, dimana data tersebut tersimpan dalam server dengan jenis basis data MySQL atau MariaDB, maka penulis perlu membuat kumpulan API. Pengguna membuat API dengan menggunakan *php framework* yaitu CodeIgniter. CodeIgniter mengimplementasi arsitektur Model-View-Controller dimana semua perintah menuju database secara langsung disimpan dalam Model dan mengolah data yang didapatkan oleh Model dengan menggunakan Controller. Data yang sudah dioleh kemudian ditampilkan melalui View.



Gambar 3.37: Bagan Alur Kerja MVC

Namun untuk membuat API, penulis tidak menggunakan View dari CodeIgniter tersebut. Penulis hanya perlu menyimpan *query* basis data pada model dan API tersimpan dalam bagian Controller. Aplikasi Android nantinya akan mengakses Controller tersebut dan dijadikan sebagai API untuk mendapatkan akses kepada basis data. API yang dibuat untuk Aplikasi Android hanya memiliki metode yang bersifat *read only*. Data akan diterima oleh aplikasi dalam bentuk JSON (JavaScript Object Notation).



Gambar 3.38: Bagan Alur Kerja CodeIgniter-Based API

Sampel kode dari CodeIgniter-Based API yang telah dibuat oleh Penulis dapat dilihat pada Gambar 3.39 atau pada Lampiran C dan D.



```

16     function get_resep(){
17         $query = $this->Masakyuk_model->get_resep();
18         $response = array();
19         $cek = $query;
20         if($cek > 0){
21             $response["resep"] = array();
22             foreach ($query as $row) {
23                 $data=array();
24                 $data["id_resep"] = $row["id_resep"];
25                 $data["id_cat"] = $row["id_cat"];
26                 $data["judul"] = $row["judul"];
27                 $data["gambar"] = $row["gambar"];
28                 $data["link_youtube"] = $row["link_youtube"];
29                 $data["rekомендasi"] = $row["rekомендasi"];
30                 array_push($response["resep"], $data);
31             }
32             $response["success"] = 1;
33             $response["message"] = "Semua data resep";
34             echo json_encode($response);
35         } else {
36             $response["success"] = 0;
37             $response["message"] = "Data resep tidak ditemukan";
38             echo json_encode($response);
39         }
40     }
41

```

Gambar 3.39: Sampel Kode CodeIgniter API-Based yang Dibuat Penulis

3.3.3 Melakukan Input Data ke Basis Data

Input data ke dalam *database* dapat dilakukan dengan dua cara, yaitu memasukkan data melalui phpMyAdmin atau memasukan data melalui aplikasi web khusus untuk memasukkan data ke dalam basis data.

The screenshot shows the phpMyAdmin interface for the 'tb_resep' table. At the top, there are tabs for Browse, Structure, SQL, Search, Insert, Export, Import, Privileges, and Operations. The 'Insert' tab is active. Below the tabs, the table structure is displayed with columns: id_resep (int(11)), id_cat (int(11)), judul (text), gambar (varchar(50)), link_youtube (varchar(20)), and rekomendasi (tinyint(1)). The 'judul' field has a large text input area. The 'gambar' and 'link_youtube' fields have dropdown menus. The 'rekomendasi' field has a dropdown menu. A 'Go' button is located at the bottom right of the form.

Gambar 3.40: Input Data dengan phpMyAdmin

Penulis mengembangkan aplikasi berbasis web dengan menggunakan *framework* CodeIgniter yang khusus untuk memasukkan data ke dalam basis data. Adanya aplikasi berbasis web tersebut memudahkan penulis dalam memasukkan data-data ke dalam basis data, mulai dari data resep, bahan, hingga data harga bahan.

The screenshot shows a web application titled 'Masak Yuk'. The main page is titled 'Daftar Resep' (Recipe List). It displays a table with columns: Judul, Kategori, Link YouTube, Rekomendasi, and Aksi. There are two rows of data: 'Siomay Jamur Masako' (Makanan Pembuka, Ya) and 'Nasi Goreng Crispy' (Makanan Utama, Belum). Each row has a set of actions: Edit, Lihat Bahan, Lihat Cara Memasak, and Hapus. A 'Tambah' button is located at the top right of the table.

Judul	Kategori	Link YouTube	Rekomendasi	Aksi
Siomay Jamur Masako	Makanan Pembuka	CXBMJMJIBSWM	Ya	Edit Lihat Bahan Lihat Cara Memasak Hapus
Nasi Goreng Crispy	Makanan Utama	QdhXqHtOPmA	Belum	Edit Lihat Bahan Lihat Cara Memasak Hapus

Gambar 3.41: Tampilan List Resep pada Aplikasi Berbasis Web

Masak Yuk

Menu ▾ Administrator ▾

Input Data Resep

Nama Resep:

Kategori: --Pilih Kategori-- ▾

Link YouTube:

Rekomendasi:

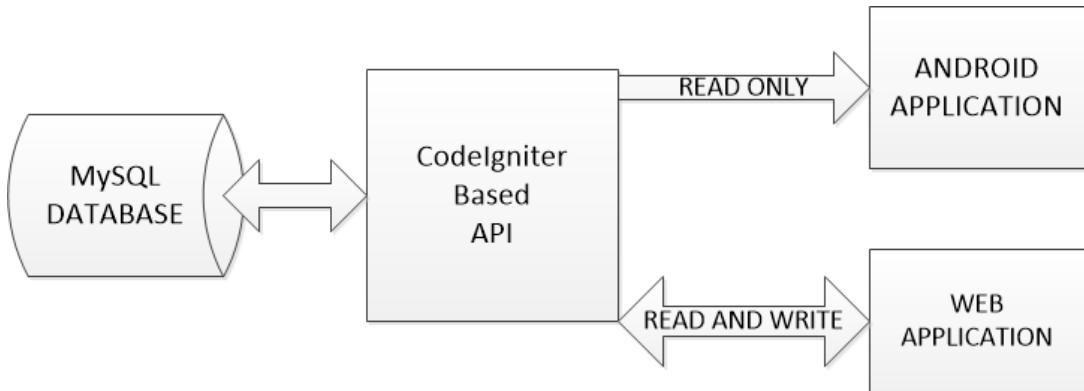
Gambar:

Choose file | No file chosen

Simpan

Gambar 3.42: Tampilan Input Resep pada Aplikasi Berbasis Web

Penulis melakukan sedikit modifikasi pada API yang telah dibuat dengan mengubah metode *read only* menjadi *read and write* dengan catatan yaitu write hanya diperbolehkan pada aplikasi berbasis web yang digunakan untuk memasukkan data-data yang dibutuhkan.



Gambar 3.43: Bagan Alur Kerja CodeIgniter-Based API Hasil Modifikasi

3.3.4 Implementasi Desain Aplikasi

Desain Aplikasi sudah diimplementasikan kedalam bentuk *layout* Android Studio yaitu dengan format xml sehingga penulis dapat melewati tahap implementasi desain ini. Implementasi dapat dilihat pada Bagian 3.2.6.

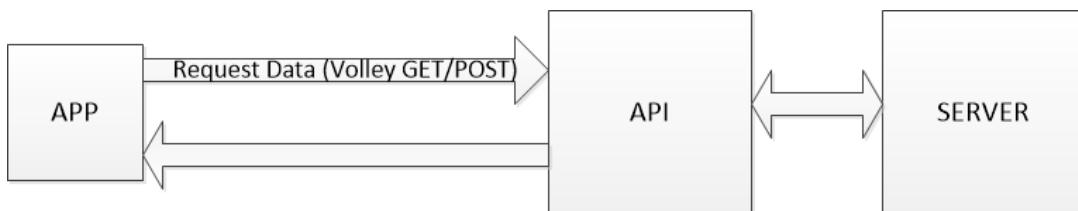
3.3.5 Implementasi Metode Pertukaran Data antara Aplikasi dengan Basis Data

Pada tahap ini, penulis mencoba mengimplementasikan API yang telah dibuat ke dalam aplikasi Android. Hal itu dilakukan agar aplikasi dapat melakukan pertukaran data dengan basis data dengan dijembatani oleh API yang telah dibuat sebelumnya. Pada aplikasi berbasis web, pertukaran data dilakukan dengan memanfaatkan Form GET/POST seperti pada Gambar 3.44.



Gambar 3.44: Bagan Alur Kerja Form GET/POST

Dalam pengembangan aplikasi Android, salah satu cara untuk dapat memanggil CodeIgniter-Based API yang telah dibuat adalah dengan menggunakan Volley. Android Volley Library adalah sebuah *library* yang disediakan oleh Google untuk dapat memfasilitasi akses *Web-Based API* pada aplikasi Android sebagai media pertukaran data antara aplikasi dengan basis data. Cara Kerja Volley secara umum menyerupai Form GET/POST pada aplikasi berbasis web.



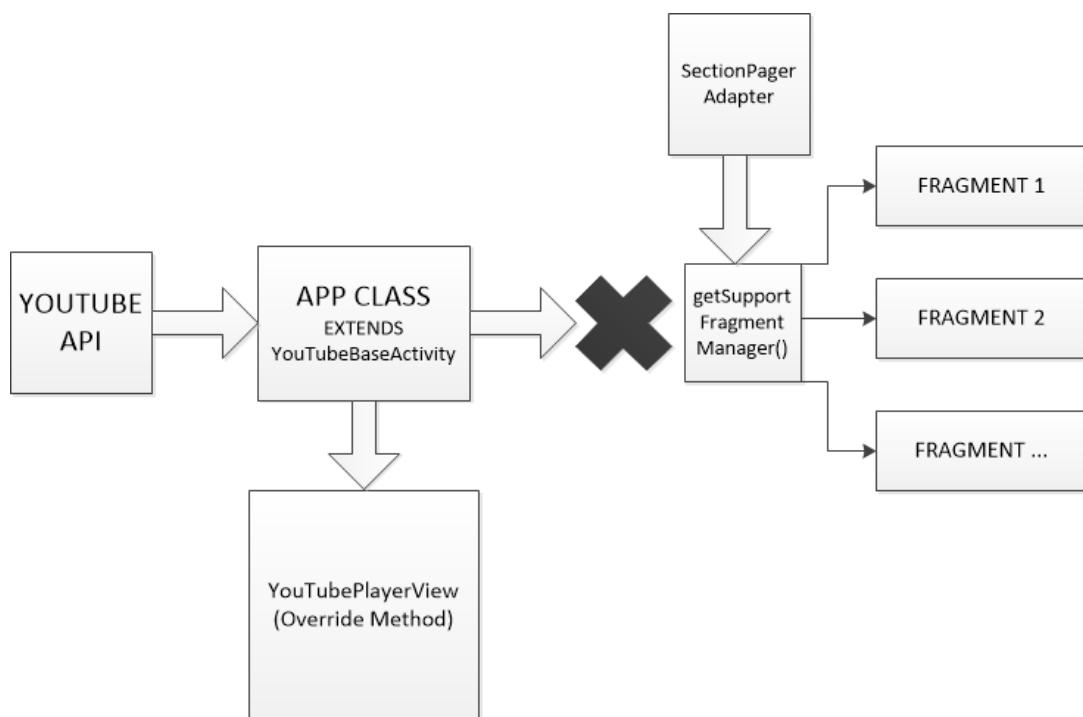
Gambar 3.45: Bagan Alur Kerja Volley

Untuk dapat menggunakan Volley, penulis harus terlebih dahulu menambahkan compile 'com.android.volley:volley:1.0.0' pada berkas build.gradle yang terdapat

di dalam direktori app.

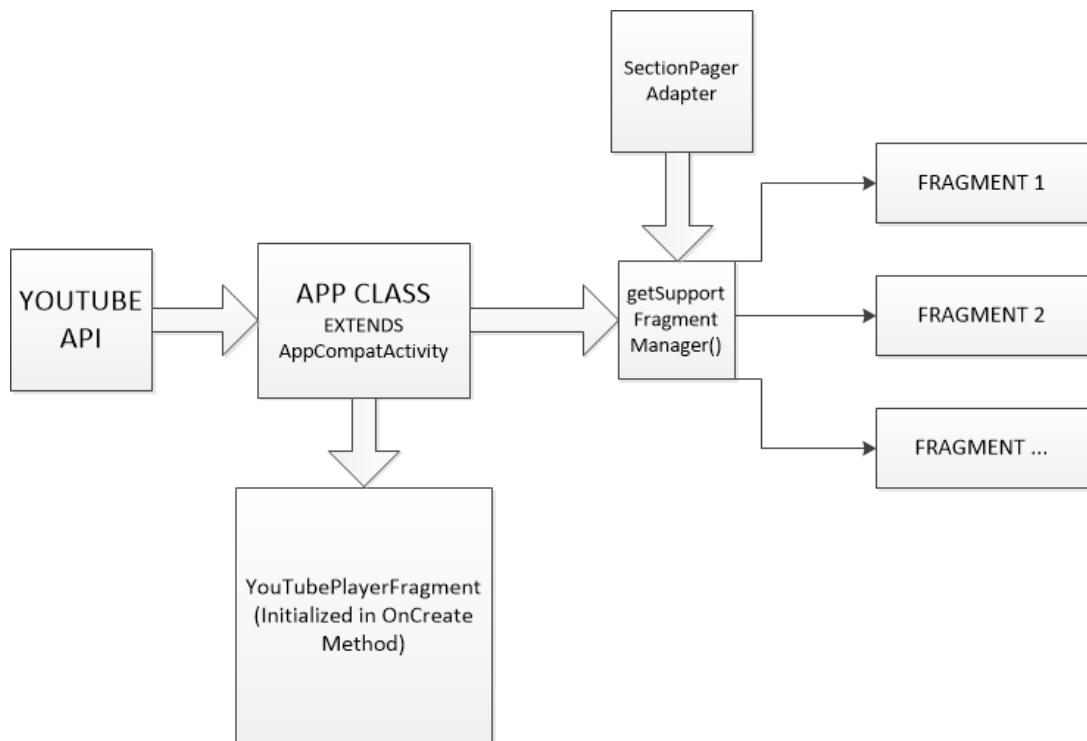
3.3.6 Mengintegrasikan YouTube dengan Aplikasi Android

Untuk dapat memainkan video YouTube pada aplikasi Android, maka dibutuhkan YouTube Android Player API sebagai perantara aplikasi dengan basis data YouTube. Penulis melakukan sedikit modifikasi pada implementasi YouTube Android Player API karena penggunaan Fragment pada Resep Activity. Ketika suatu *class* pada Java Android Studio menggunakan Fragment, maka diperlukan *class* yang melakukan ekstensi (*extends*) dengan AppCompatActivity untuk mendukung pengeleolaan beberapa Fragment yang akan ditampilkan pada sebuah Activity. Sedangkan YouTube Android Player API mewajibkan penggunanya untuk melakukan ekstensi terhadap *class* YouTubeBaseActivity



Gambar 3.46: Bagan Alur Kerja YouTubePlayerView pada sebuah Activity

Apabila Activity tetap mengekstensi YouTubeBaseActivity, maka Activity tersebut tidak dapat mengelola Fragment yang terdapat didalamnya karena kehilangan metode getSupportFragmentManager() yang berasal dari ekstensi *class* terhadap AppCompatActivity. Maka dari itu, penulis melakukan modifikasi dengan tidak menggunakan YouTubePlayerView, melainkan menggunakan YouTubePlayerFragment. Keputusan tersebut diambil karena YouTubePlayerFragment "lebih ramah" terhadap Activity yang memiliki Fragment dengan tidak perlu mengekstensi *class* YouTubeBaseActivity sehingga seluruh Fragment yang ada pada sebuah Activity dapat dikelola dengan baik. Perbedaannya adalah seluruh metode yang diberikan oleh YouTube Android Player API tidak dilakukan *override*, melainkan langsung diinisialisasikan pada metode onCreate sebuah Activity.



Gambar 3.47: Bagan Alur Kerja YouTubePlayerFragment pada sebuah Activity

Sampel kode implementasi YouTubePlayerFragment terdapat pada Gambar 3.48.

```

YouTubePlayerFragment youtubeFragment = (YouTubePlayerFragment)
        getSupportFragmentManager().findFragmentById(R.id.youtube_view);
youtubeFragment.initialize(YOUTUBE_API_KEY,
        new YouTubePlayer.OnInitializedListener() {
            @Override
            public void onInitializationSuccess(YouTubePlayer.Provider provider,
                                                YouTubePlayer youtubePlayer, boolean b) {
                if(!b){
                    youtubePlayer.cueVideo(link_youtube);
                }
            }
            @Override
            public void onInitializationFailure(YouTubePlayer.Provider provider,
                                                YouTubeInitializationResult errorReason) {
                if (errorReason.isUserRecoverableError()) {
                    errorReason.getErrorDialog(getParent(), RECOVERY_REQUEST).show();
                } else {
                    String error = String.format("Error initializing YouTube player: %s", errorReason.toString());
                    Toast.makeText(getApplicationContext(), error, Toast.LENGTH_LONG).show();
                }
            }
        });
    });
}

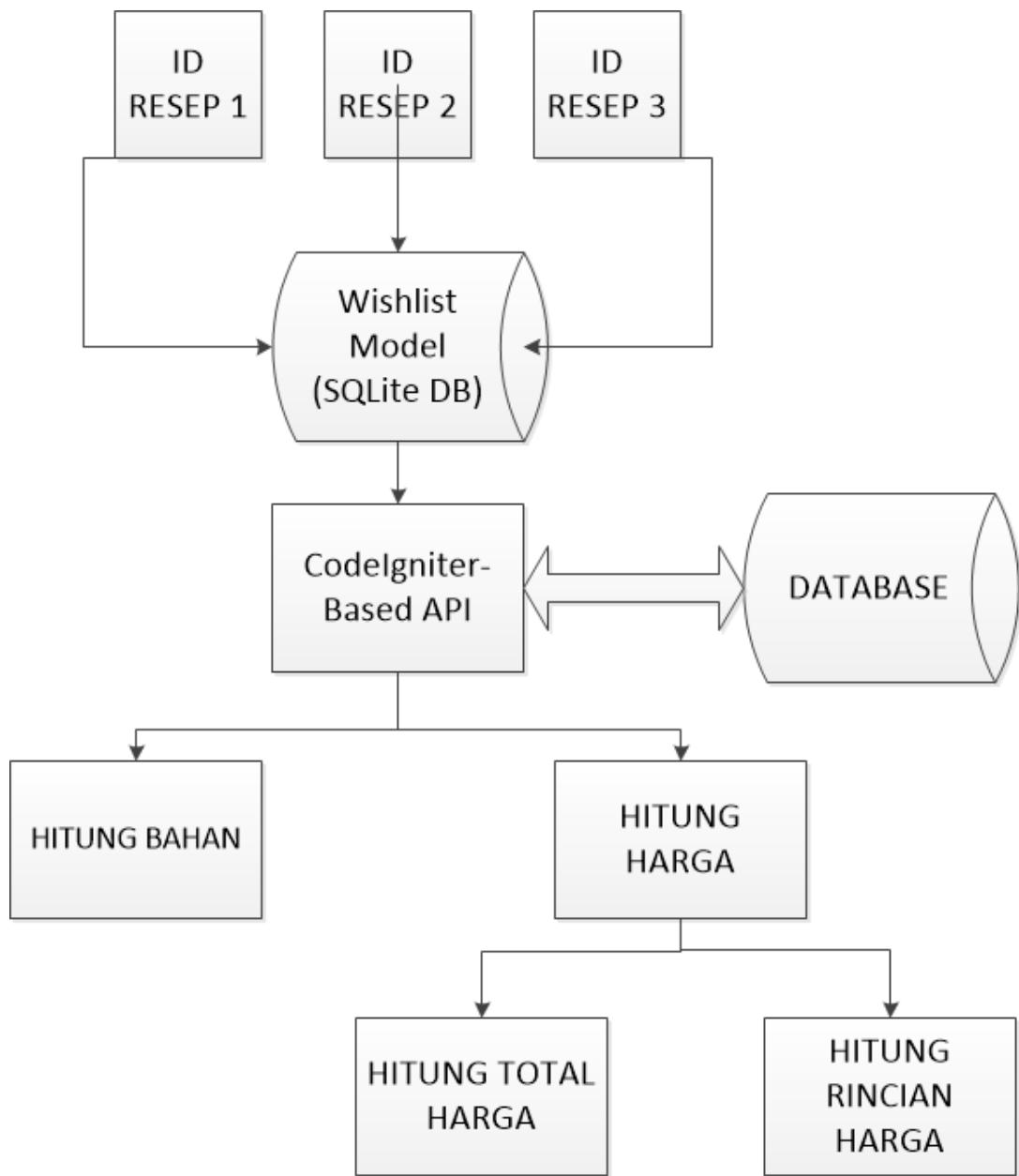
```

Gambar 3.48: Sampel Kode Implementasi YouTubePlayerFragment pada Metode onCreate sebuah Activity

3.3.7 Mengimplementasi Fitur Lainnya

Fitur lain yang diimplementasikan oleh penulis adalah fitur penyimpanan *wishlist*, menghitung total bahan pada resep yang terdapat dalam *wishlist*, menghitung total harga bahan yang terdapat pada *wishlist* dan menampilkan rincian harga per bahan. *Wishlist* sendiri dibuat dengan memanfaatkan basis data SQLite dengan menggunakan bahasa query yang sama yakni SQL. Basis data SQLite bersifat lokal dan terdapat didalam aplikasi itu sendiri. Jadi, apabila pengguna aplikasi menghapus data aplikasi pada ponselnya, maka data *wishlist* pun juga akan hilang.

Wishlist mampu menampung maksimal tiga resep masakan. Untuk dapat menghitung total bahan dan total harga bahan yang terdapat pada resep-resep tersebut, maka data yang terdapat di dalam SQLite dikirimkan dan diolah oleh API yang telah dibuat untuk kemudian disalurkan kembali hasil perhitungannya pada aplikasi. Alur dari proses penyimpanan pada *wishlist* hingga perhitungan total bahan dan harga bahan dijelaskan pada Gambar 3.49



Gambar 3.49: Alur Penyimpanan pada SQLite dan Pengolahan Total Bahan dan Harga Bahan

DAFTAR PUSTAKA

- [1] Anisya. 2013. "Aplikasi Sistem Database Rumah Sakit Terpusat pada Rumah Sakit Umum (RSU) Aisyiyah Padang dengan Menerapkan Open Source (PHP - MySQL)". Jurnal Momentum.
- [2] Ardi. 2013. "Pengembangan Sistem Pemesanan Makanan Berbasis Android Menggunakan Troli Cerdas". Skripsi Sarjana pada FT Universitas Indonesia.
- [3] Artikel non-personal. 2016. "SDLC - Spiral Model". Tutorials Point [Online]. Tersedia: https://www.tutorialspoint.com/sdlc/sdlc_spiral_model.htm. [28 Oktober 2016].
- [4] Bagui, Sikha dan Richard Earp. 2011. *Database Design Using Entity-Relationship Diagram*. Florida: CRC Press.
- [5] Bell, Donald. 2003. "An Introduction to the Unified Modeling Language". IBM developerWorks [Online]. Tersedia: <http://www.ibm.com/developerworks/rational/library/769.html>. [01 November 2016].
- [6] Burton, Michael dan Donn Felker. 2012. *Android Application Development For Dummies, 2nd Edition*. New Jersey: John Wiley and Sons, Inc.
- [7] Denis, Alan, Barbara Haley Wixom dan David Tegarden. 2010. *Systems Analysis and Design with UML: An Object-Oriented Approach 3rd Edition*. New Jersey: John Wiley and Sons, Inc.
- [8] Edi, Doro dan Stevalin Betshani. 2009. "Analisis Data dengan Menggunakan ERD dan Model Konseptual Data Warehouse". Jurnal Informatika, Vol.5, No. 1, Juni 2009.

- [9] Francis, Danny. 2013. "Perancangan Aplikasi Customer Relationship Management: Studi Kasus PT Primacipta Megah Jaya". Thesis Magister pada FASILKOM Universitas Indonesia.
- [10] Lee, Sunguk. 2012. "Unified Modeling Language (UML) for Database Systems and Computer Application". International Journal of Database Theory and Application.
- [11] Murtiwiyat dan Glenn Lauren. 2013. "Rancang Bangun Aplikasi Pembelajaran Budaya Indonesia Untuk Anak Sekolah Dasar Berbasis Android". Jurnal Ilmiah KOMPUTASI, Volume 12 Nomor : 2, Desember 2013.
- [12] Myers, Brad A dan Jeffrey Stylos. 2016. "Improving API Usability". Communications of the ACM, vol 59, No. 6, June, 2016.
- [13] Palupi, Sri dan Tuti Hera Widi Handayani. 2010. *Resep Masakan (Modul 2)*. Yogyakarta: Universitas Negeri Yogyakarta.
- [14] Pertiwi, Ratih Sukma dan Annelis Brillian. 2015. "Perempuan, Mari Kembali ke Dapur!". Tabloid Nova [Online]. Tersedia: <http://tabloidnova.com/Sedap/Tips-Masak/Perempuan-Mari-Kembali-Ke-Dapur>. [28 April 2016].
- [15] Pressman, Roger S. 2010. *Software Engineering A Practitioner's Approach*. New York: McGraw Hill.
- [16] Rogers, Rick, dkk. 2009. *Android Application Development*. California: O'Reilly Media.
- [17] Setyanti, Christina Andhika dan Syafrina Syaaf. 2014. "3 Alasan Utama Perempuan Malas Masak". Kompas.com [Online]. Tersedia:

- <http://female.kompas.com/read/2014/03/06/1254310/3.Alasan.Utama.Perempuan.Malas.Masak>. [28 April 2016].
- [18] Soenardi, Tuti dan Tim. 2013. *Teori Dasar Kuliner*. Jakarta: PT Gramedia Pustaka Utama.