

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada era digital sekarang ini, internet semakin mudah dijangkau seiring dengan semakin meningkatnya pengguna internet dan *smartphone* di Indonesia. Berbagai aspek kehidupan tidak terlepas dari internet membuat perubahan pada perilaku belanja konsumen. Menurut artikel dari (Media Konsumen, 2016), berdasarkan hasil survei di India yang dilakukan oleh couponrani.com dapat memberi gambaran bagaimana era digital telah mempengaruhi perilaku masyarakat dalam berbelanja. Meskipun survei tersebut dilakukan di India, secara umum perilaku konsumen Indonesia di era digital memiliki kecenderungan yang serupa. Hasil survei tersebut menunjukkan 92% konsumen memanfaatkan internet untuk mencari informasi produk sebelum melakukan keputusan pembelian di toko konvensional (*offline*). Sebanyak 54% dari konsumen memanfaatkan internet untuk membandingkan harga, 18% membandingkan produk yang diminati dengan produk sejenis yang lain, dan 20% membaca *review* atau *feedback* produk.

Kebutuhan untuk membandingkan harga produk dari satu tempat belanja dengan tempat belanja yang lain dikarenakan perilaku konsumen yang menginginkan produk dengan harga yang murah. Pada saat berbelanja jika harga suatu produk di toko yang sedang dikunjungi mahal, konsumen akan menunda membeli beberapa barang untuk dibeli di toko lain yang menjual produk tersebut dengan harga yang lebih murah. Kondisi ini menyebabkan belanja konsumen tidak efisien. Jika konsumen tetap membeli barang tersebut di toko yang sama dengan kondisi harga produk yang lebih mahal, maka akan menyebabkan ketidakpuasan tersendiri dari konsumen.

Namun ketika membandingkan harga produk ditemui kendala untuk mendapatkan informasi harga produk pada toko konvensional. Saat ini sudah ada beberapa aplikasi yang menyediakan perbandingan harga produk seperti priceza, pricebook, dan pricearea. Namun aplikasi yang sudah ada tersebut mengandalkan harga produk yang ada pada toko-toko online. Selain itu, tidak semua toko bersedia memberikan informasi harga produknya. Hanya toko ritel berskala besar yang

bersedia memberikan informasi harga produknya dalam bentuk katalog, sedangkan minimarket atau supermarket lokal hanya menyediakan label harga pada rak tokonya. Harga produk yang disediakan pada katalog juga hanya produk yang sedang promo atau diskon saja sedangkan produk lain yang sedang tidak diskon tidak ditampilkan. Selain itu harga produk pada katalog mempunyai periode waktu tertentu sehingga informasi akan usang. Kurangnya informasi harga produk menyebabkan konsumen sulit untuk membandingkan harga suatu produk pada satu tempat belanja dengan tempat belanja yang lainnya.

Selain itu, harga produk yang selalu berubah-ubah menjadi acuan diperlukannya sebuah aplikasi yang dapat melakukan pembaruan data. Salah satu metode yang dapat dipakai untuk mendapat informasi harga terbaru suatu produk yaitu melalui partisipasi masyarakat. Partisipasi masyarakat dalam penelitian ini digunakan sebagai konsep dalam membangun aplikasi sehingga aplikasi yang dihasilkan dapat menghimpun data dari pengguna. Partisipasi masyarakat digunakan karena tidak ada toko konvensional yang bersedia memberikan informasi harga produknya sehingga masyarakat sendirilah yang harus memberikan data. Dalam menghimpun data dari masyarakat, diperlukan sebuah cara agar masyarakat memberikan data harga yang benar yaitu dengan membangun sistem rekomendasi layanan harga produk terendah ini dalam bentuk sebuah aplikasi catatan belanja.

Selain digunakan agar masyarakat memberikan data yang benar, catatan belanja dibangun dengan menambahkan fitur pengeluaran berdasarkan catatan belanja yang telah diinputkan. Fitur ini merupakan sebuah cara untuk membuat aplikasi ini dibutuhkan oleh masyarakat agar masyarakat tertarik dan dengan sukarela mencatat belanjaannya sehingga tanpa sadar masyarakat ikut berpartisipasi dalam memberikan informasi harga produk. Jika banyak masyarakat yang berpartisipasi melakukan pencatatan belanjanya pada aplikasi maka informasi harga yang didapatkan dari aplikasi bisa menjadi harga produk yang *ter-update*. Melalui catatan belanja yang dikumpulkan dari banyak pengguna maka akan dapat dibandingkan harga produk dari satu tempat belanja dengan tempat belanja lain yang dapat membantu pengguna mendapatkan rekomendasi tempat belanja dengan harga produk yang terendah. Selain itu, dapat diketahui harga-harga produk

sehingga konsumen dapat membuat rencana belanja dengan perkiraan jumlah uang yang akan dikeluarkan pada saat berbelanja.

Berdasarkan penjelasan yang telah dipaparkan, maka perlu dilakukan pengembangan aplikasi catatan belanja yang dapat membantu masyarakat mendapatkan informasi harga produk terendah dengan dukungan partisipasi masyarakat.

1.2 Perumusan Masalah

Kebutuhan konsumen untuk mengetahui harga terendah suatu produk dikarenakan perilaku masyarakat yang menginginkan pembelian produk dengan harga terendah. Informasi harga suatu produk dapat mempengaruhi tempat belanja yang akan dituju oleh masyarakat. Pada saat berbelanja jika harga suatu produk di toko yang sedang dikunjungi mahal, konsumen akan menunda membeli beberapa barang untuk dibeli di toko lain yang menjual produk tersebut dengan harga yang lebih murah.

Melalui catatan belanja, konsumen akan dapat mengetahui harga produk serta membandingkan harga produk karena konsumen mencatat semua produk yang pernah dibelinya beserta keterangan harga produk tersebut. Namun jika hanya berdasarkan catatan pribadi, konsumen hanya akan dapat membandingkan harga dengan toko-toko yang pernah dikunjunginya saja, sedangkan bisa saja toko lain yang belum pernah dikunjungi menyediakan produk yang sama dengan harga yang lebih rendah.

Maka dari itu diperlukan partisipasi dari masyarakat untuk mendapatkan informasi harga produk dari berbagai toko. Berdasarkan catatan keuangan belanja dari banyak pengguna akan didapatkan rekomendasi harga produk terendah dengan partisipasi masyarakat agar data harga yang ada merupakan data harga yang *ter-update*. Data harga produk akan dikumpulkan dari catatan belanja pengguna. Kemudian harga produk pada tempat belanja yang sama akan dipilih harga yang terbaru dan dibandingkan dengan toko lain di kota yang sama untuk kemudian direkomendasikan kepada pengguna.

Berdasarkan hal di atas yang telah disampaikan, maka permasalahan yang akan dibahas adalah bagaimana membangun aplikasi catatan belanja dengan layanan rekomendasi harga produk terendah dengan partisipasi masyarakat.

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian yang dilakukan adalah menghasilkan sistem yang dapat merekomendasikan harga produk terendah dari berbagai tempat belanja beserta keterangan tempat belanja yang menjual produk tersebut berdasarkan data harga produk yang dihimpun dari catatan belanja pengguna.

1.4 Pembatasan Masalah

Pembatasan masalah dari penelitian yang akan dilakukan adalah :

1. Aplikasi yang dibangun berbasis android.
2. Catatan belanja yang dibuat oleh pengguna memuat harga produk, deskripsi produk, serta waktu dan lokasi tempat belanja.
3. Rekomendasi yang dihasilkan tidak sampai pada proses pengambilan keputusan dan merupakan perbandingan total harga dari berbagai toko.
4. Produk yang dapat diinputkan hanya produk kebutuhan sehari-hari yang memiliki *barcode*.
5. Keterangan toko yang dapat *diinput* oleh pengguna sebagai tempat belanja hanya toko yang terdaftar dalam Google Maps.

1.5 Sistematika Penulisan

Adapun sistematika dari penulisan tugas akhir ini disusun dalam lima bab yang terdiri dari Bab I Pendahuluan, Bab II Tinjauan Pustaka, Bab III Metodologi Penelitian dan Perancangan Sistem, Bab IV Hasil Perancangan dan Analisis Sistem, serta Bab V Penutup.

Bab I Pendahuluan adalah bab yang berisi latar belakang, perumusan masalah, tujuan penelitian, pembatasan masalah, dan sistematika penulisan.

Bab II Tinjauan Pustaka adalah bab yang berisi landasan teori berkaitan dengan penelitian yang akan dilakukan. Beberapa teori-teori yang terkait adalah pengertian aktivitas anak, monitoring, evaluasi, definisi Android, Android Studio, *Unified Modeling Language* (UML), Rest API, PHP, MySQL.

Bab III Metodologi Penelitian dan Perancangan Sistem adalah bab yang berisi tentang Bahan Penelitian, Alat yang Dipergunakan, Metode Penelitian, Variabel atau Data, Diagram Alir Penelitian, Diagram Alir Sistem, Perancangan Aplikasi, Perancangan UML, Perancangan Basis Data, Perancangan Antarmuka, serta Rencana Pengujian dengan menggunakan metode blackbox dan pengujian

kuesioner.

Bab IV Implementasi dan Hasil Pengujian adalah bab yang berisi penjelasan mengenai implementasi pada sistem, screenshoot tampilan antarmuka sistem yang sudah jadi, serta analisis hasil uji coba. Setiap bagian sistem yang ditampilkan akan dilakukan analisis terlebih dahulu untuk mengarah kepada suatu kesimpulan.

Bab V Penutup adalah bab yang berisi kesimpulan dari penelitian yang telah dilakukan dan saran/rekomendasi untuk perbaikan, pengembangan atau kesempurnaan / kelengkapan penelitian yang telah dilakukan.

BAB II

TINJAUAN PUSTAKA

2.1 Catatan Belanja

Catatan belanja merupakan daftar barang yang ingin dibeli ketika akan pergi berbelanja yang ditulis pada sebuah kertas (Collinsdictionary.com, 2018). Catatan belanja memiliki berbagai fungsi. Salah satu fungsi catatan belanja yaitu dapat digunakan sebagai mengatur anggaran belanja, atau untuk perencanaan belanja. Catatan belanja juga dapat menunjukkan kebutuhan atau ketertarikan konsumen (Nurmi, Forsblom, & Floréen, 2009).

Banyak *developer* yang sudah mengembangkan catatan belanja menjadi aplikasi pada *smartphone* untuk memudahkan masyarakat dalam berbelanja. Berikut ini merupakan beberapa aplikasi catatan belanja yang ada pada *Google Play Store*:

1. *Grocery Shopping List*

Aplikasi bernama *Grocery Shopping List* ini dikembangkan oleh Listonic merupakan aplikasi untuk mencatat daftar belanjaan dengan praktis dan mudah digunakan. Pengguna dapat menambahkan jumlah item dengan mudah atau dapat memilih opsi input dengan menggunakan suara. Selain itu aplikasi ini dapat menghitung total belanja secara otomatis dan pengguna dapat berbagi daftar belanjanya dengan orang lain.

2. *Bring! Grocery Shopping List*

Bring! Grocery Shopping List merupakan aplikasi untuk merencanakan dan mengelola belanja kebutuhan sehari-hari yang sudah diunduh oleh lebih dari dua juta orang dari seluruh dunia. Pada aplikasi ini pengguna dapat membuat dan membagikan daftar belanjanya kepada keluarga dan teman-teman. Aplikasi ini dapat digunakan pada *smartphone*, *tablet*, dan *smartwatches*. Pengguna dapat menambahkan foto produk pada item belanja agar pengguna dapat selalu membeli produk dengan *brand* yang benar.

3. *Our Groceries Shopping List*

Aplikasi ini merupakan aplikasi catatan belanja yang dikembangkan oleh HeadCode. Pada aplikasi ini pengguna dapat membagikan daftar belanjanya dengan keluarga. Selain daftar belanja, pengguna juga dapat membagikan resep masakan

dan menambahkannya ke daftar belanja. Aplikasi ini juga dilengkapi dengan fitur menambahkan foto pada item belanja dan menambahkan item dengan *scan barcode*.

2.2 Partisipasi Masyarakat

Dilihat dari segi etimologi, kata partisipasi berasal dari bahasa Inggris *participate* yang artinya mengikutsertakan, ikut mengambil bagian (Wijaya, 2004). Partisipasi juga berarti peran serta seseorang atau kelompok masyarakat secara aktif dari proses perumusan kebutuhan, perencanaan, sampai pada tahap pelaksanaan kegiatan baik melalui pikiran atau langsung dalam bentuk fisik (Slamet, 1994). Partisipasi masyarakat pada penelitian ini mengarah pada partisipasi atau keikutsertaan masyarakat untuk membantu pengumpulan data barang belanjaan masyarakat berdasarkan catatan belanja yang berguna untuk pengembangan aplikasi yang dikembangkan.

Dalam penelitian ini masyarakat turut berpartisipasi dengan melakukan pencatatan belanjaannya dari data struk belanja. Dari catatan belanja masyarakat, dapat diketahui harga produk dari toko-toko yang dikunjungi masyarakat. Informasi mengenai harga produk yang tidak statis dan dapat mengalami perubahan harga dan diskon barang di toko tertentu menjadi informasi yang sangat dibutuhkan konsumen ketika merencanakan belanjanya. Informasi mengenai harga dan diskon produk diperlukan melalui partisipasi masyarakat untuk membantu menjadikan aplikasi yang dikembangkan peneliti menjadi aplikasi dengan harga produk yang selalu ter-*update* dan dapat membantu memberikan informasi yang jelas dan berguna untuk konsumen. Selain itu, data yang didapatkan dari partisipasi masyarakat dapat diolah menjadi acuan untuk melihat produk yang *booming* sebagai informasi bisnis untuk produsen dan toko-toko.

Salah satu aplikasi yang memanfaatkan partisipasi masyarakat adalah Snapcart. Snapcart adalah aplikasi *mobile* yang memungkinkan penggunanya mendapatkan *cashback* dari foto struk belanjaannya. Aplikasi ini dapat dimanfaatkan oleh *brand* barang konsumsi untuk memonitor perilaku konsumsi secara langsung karena struk belanja yang didapatkan oleh Snapcart merupakan data *real-time*. Cara yang digunakan Snapcart menarik masyarakat agar menggunakan aplikasinya serta memberikan data struk belanjanya yaitu dengan

memberikan *cashback* mulai dari Rp 150 hingga Rp 23.350. Snapcart hanya mengumpulkan data struk belanja konsumen, dan belum membantu memberikan timbal balik informasi harga produk yang dapat menjadi acuan untuk pembelanjaan barang masyarakat pada rencana belanja berikutnya. Data yang dikumpulkan Snapcart digunakan untuk memonitor perilaku konsumsi secara langsung berdasarkan rekaman struk belanja masyarakat.

2.3 Android

Android merupakan sebuah sistem operasi telepon seluler dan komputer tablet layar sentuh (*touchscreen*) yang berbasis Linux. Android menggunakan OS kernel Linux, UI yang berkualitas, *end-user application*, *code libraries*, *framework* aplikasi, mendukung multimedia dan banyak lagi. Aplikasi yang dibuat untuk Android menggunakan bahasa pemrograman Java.

Ada dua cara untuk membangun atau membuat aplikasi berbasis android, yaitu memiliki perangkat telepon seluler yang berbasis android langsung dan menggunakan emulator yang sudah disediakan oleh Google. Sebelum memulai membangun aplikasi berbasis android, diperlukan beberapa perangkat, antara lain :

- *Android Studio*
- *Sun's Java Development Kit (JDK)*.
- *The Android Software Developer's Kit (SDK)*.

Windows, Linux, dan Mac OS X merupakan sistem operasi yang dapat digunakan untuk pengembangan pembuatan aplikasi berbasis Android dengan memanfaatkan Android SDK (Elian, S., & Djanali, 2012).

Pada penelitian ini, dilakukan rekomendasi harga produk terendah dengan menggunakan Android Studio. Android pada penelitian ini berfungsi sebagai sistem operasi yang akan menampilkan data dari *mysql* dengan *php* dengan format *JSON* (*JavaScript Object Notation*).

2.4 Barcode Scanner

Barcode Scanner adalah suatu alat yang digunakan untuk membaca kode-kode bar. Kode bar terdapat pada tanda pengenal produk. Penggunaan *barcode scanner* dapat memperkecil kesalahan input yang disebabkan kesalahan operator komputer atau kasir. Penggunaan *barcode scanner* juga dapat mempercepat proses

memasukkan data (Soleh, Sopiyan, Ristiandana, & Zaeni, 2013). Pada penelitian ini *barcode scanner* digunakan dengan tujuan agar pengguna tidak perlu mengetik nama produk ketika menginput data struk belanja pada aplikasi.

Berikut merupakan jenis *barcode* yang standar dan diakui untuk retail, yaitu :

1. EAN-13

EAN (*European Article Numbering*) diimplementasikan oleh *International Article Numbering Association* di Eropa. EAN-13 digunakan sebagai standar kode batang retail di Eropa dan seluruh dunia kecuali Amerika dan Kanada. Standar EAN-13 terdiri dari kode negara atau kode sistem yang menunjukkan negara dimana *manufacturer* terdaftar yang terletak pada 3 digit pertama kode batang, *manufacturer code*, kode produk, dan *check digit* atau *checksum* (Saghranie Daulay, 2010). Berikut merupakan contoh barcode EAN-13 pada gambar 2.1.



Gambar 2.1 Contoh barcode EAN-13

2. EAN-8

EAN-8 (*European Article Numbering*) merupakan versi yang dipendekan dari versi kode EAN-13. EAN-8 terdiri atas 2 atau 3 digit kode negara , 4 atau 5 digit data (tergantung panjang kode negara) dan sebuah digit cek. Kode EAN-8 dikhususkan untuk mengidentifikasi produk dan pembuat produk.



Gambar 2.2 Contoh barcode EAN-8

3. UPC (*Universal Product Code*)

UPC (*Universal Product Code*) merupakan sebuah *barcode numeric* dan memiliki panjang baris yang tetap (fixed). UPC digunakan secara luas pada industri grosir, khususnya di Amerika Serikat dan Kanada untuk pelabelan pada produk-produk berukuran kecil/ritel (Saghranie Daulay, 2010). Standar kode barisnya, yaitu UPC-A, terdiri atas 1 digit nomor sistem pada awal kode baris, 5 digit nomor manufaktur, 5 digit nomor produk, serta 1 digit cek. Berikut merupakan contoh

barcode UPC.



Gambar 2.3 Contoh *barcode* UPC-A

Pada android studio untuk menambahkan *barcode scanner* pada aplikasi yang dibangun menggunakan *library* dari Avaneesh Maurya yang dibangun dengan menggunakan barcode detector pada Google Mobile Vision API. Barcode detector pada Mobile Vision API dapat mendeteksi *barcode* secara *real time* dalam segala orientasi. Format barcode yang dapat dibaca oleh API yaitu EAN-13, EAN-8, Code-39, Code-93, Code-128, UPC-A, UPC-E, ITF, dan Codabar.

Barcode scanner ditambahkan dengan cara memasukkan *library barcode scanner* pada *project*. Kemudian karena *library* yang digunakan memerlukan Google Mobile Vision API maka perlu ditambahkan terlebih dahulu dengan cara menambahkan *dependency* pada *project* aplikasi seperti berikut (Maurya, n.d.).

Kode Program 2.1 Menambahkan *dependency* pada *project*.

```
1. dependencies {
2.     implementation 'com.google.android.gms:play-services-vision:15.0.2'
3. }
```

Barcode scanner dapat ditampilkan pada halaman lain dengan memanggil metode berikut.

Kode Program 2.2 Kode program menampilkan *barcode scanner*

```
1. Intent launchIntent = BarcodeReaderActivity.getLaunchIntent(this,
    true, false);
2. startActivityForResult(launchIntent,
    BARCODE_READER_ACTIVITY_REQUEST);
```

Setelah *barcode scanner* ditampilkan, *barcode scanner* akan membaca *barcode* dengan format yang sesuai yang dapat dibaca dan kemudian akan mengirimkan hasil *scan* yang dapat diterima dan diambil nilainya dengan cara sebagai berikut.

Kode Program 2.3 Kode program menerima dan mengambil nilai *barcode*

```

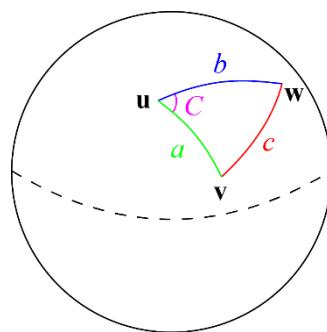
1. protected void onActivityResult(int requestCode, int resultCode,
   Intent data) {
2.     super.onActivityResult(requestCode, resultCode, data);

3.     if (resultCode != Activity.RESULT_OK) {
4.         Toast.makeText(this, "error in scanning",
5.                     Toast.LENGTH_SHORT).show();
6.         return;
7.     }

8.     if (requestCode == BARCODE_READER_ACTIVITY_REQUEST && data != null)
{
9.         Barcode barcode =
10.            data.getParcelableExtra(BarcodeReaderActivity.KEY_CAPTURED_BARCO
DE);
11.        Toast.makeText(this, barcode.rawValue,
12.                      Toast.LENGTH_SHORT).show();
13.    }
14. }
15. }
```

2.5 Formula Haversine

Formula Haversine adalah persamaan yang digunakan dalam navigasi. Formula Haversine merupakan suatu metode untuk mengetahui jarak antar dua titik dengan memperhitungkan bahwa bumi bukanlah sebuah bidang datar namun seperti hukum haversine yang berdasarkan bentuk bumi yang bulat (*spherical earth*) dengan menghilangkan faktor bahwa bumi sedikit elips (*ellipsoidal factor*). Formula ini merupakan bentuk persamaan khusus trigonometri bola, hukum haversine, yang mencari hubungan sisi dan sudut pada segitiga dalam bidang bola (Setiawan, 2014).



Gambar 2.4 Ilustrasi *Spherical law of cosines*

Dalam unit bola, sebuah “segitiga” pada permukaan bola didefinisikan sebagai lingkaran-lingkaran besar yang menghubungkan tiga poin u, v, dan w pada bola. Jika panjang dari ketiga sisi adalah (dari u ke v), b (dari u ke w), dan c (dari v

ke w), dan sudut sudut yang berlawanan c adalah C, maka hukum haversines menjadi:

$$\text{haversin}(c) = \text{haversin}(a - b) + \sin(a) \sin(b) \text{haversin}(C)$$

Formula Haversine masih dianggap baik untuk perhitungan numeris, bahkan untuk permukaan yang kecil. Berikut merupakan Rumus Haversine yang dipublikasikan oleh Roger Sinnott pada majalah Sky & Telescope pada tahun 1984 (“*Virtues of the Haversine*”) (Veness, 2002).

$$\Delta\text{lat} = \text{lat2}-\text{lat1}$$

$$\Delta\text{lng} = \text{lng2}-\text{lng1}$$

$$a = \sin^2(\Delta\text{lat}/2) + \cos(\text{lat1}) * \cos(\text{lat2}) * \sin^2(\Delta\text{lng}/2)$$

$$c = 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a})$$

$$d = R \cdot c$$

Keterangan :

R = jari-jari bumi sebesar 6371(km)

Δlat = besaran perubahan *latitude*

Δlng = besaran perubahan *longitude*

c = kalkulasi perpotongan sumbu

d = jarak (km)

2.6 Google Places API

Google Places API memungkinkan pengguna untuk mengirimkan query untuk informasi tempat pada berbagai kategori, seperti perusahaan, lokasi geografis, dan banyak lagi. Layanan ini menggunakan *HTTP requests* untuk memberikan informasi berbagai tempat. *Google Places API* memungkinkan untuk melakukan pencarian tempat terdekat maupun pencarian berdasarkan teks (Google Developers, 2014).

2.6.1 Permintaan Pencarian Terdekat

Pencarian terdekat memungkinkan pengguna mencari tempat dalam area tertentu. Pengguna dapat menyesuaikan permintaan pencarian dengan memberikan kata kunci atau menentukan jenis tempat yang dicari. Permintaan pencarian terdekat merupakan URL HTTP dengan bentuk berikut.

```
https://maps.googleapis.com/maps/api/place/nearbysearch/output
?parameters
```

Output dari bentuk tersebut dapat berupa salah satu dari nilai JSON (dianjurkan) menunjukkan *output* dalam *JavaScript Object Nation* (JSON) atau XML yang menunjukkan *output* XML.

2.6.2 Parameter Permintaan Google Places API

Parameter tertentu yang diperlukan untuk memulai permintaan pencarian terdekat. Seperti standar pada URL, semua parameter dipisah menggunakan karakter ampersand (&). Berikut merupakan parameter permintaan *Google Places API* (Google Developers, 2014) pada tabel 2.1 .

Tabel 2.1 Parameter permintaan *Google Places API*

Nama Parameter	Keterangan	Jenis
key	API Key ini mengidentifikasi aplikasi untuk keperluan manajemen kuota, sehingga tempat yang ditambahkan dari aplikasi yang dibuat langsung tersedia untuk aplikasi tersebut.	Utama
location	Garis lintang atau bujur sekitar lokasi yang ingin diambil informasinya. Parameter ini harus ditetapkan sebagai lintang, bujur.	Utama
radius	Mendefinisikan jarak (dalam meter) untuk menghasilkan pencarian tempat. Jarak maksimum yang diperbolehkan adalah 50.000 meter.	Utama
keyword	Istilah yang akan dicocokkan terhadap semua materi yang terindeks oleh Google untuk suatu tempat, tidak terbatas pada nama, jenis, alamat, serta ulasan pelanggan dan materi pihak ketiga lainnya.	Tambahan

Nama Parameter	Keterangan	Jenis
language	Kode bahasa, menunjukkan dalam bahasa apa hasil permintaan akan dikembalikan, jika memungkinkan.	Tambahan
opennow	Hanya mengembalikan tempat yang sedang buka pada saat kueri dikirim. Tempat yang tidak menetapkan jam buka dalam database Google Places tidak akan dikembalikan jika menggunakan ini dalam kueri	Tambahan
types	Membatasi hasil ke tempat yang cocok dengan tipe yang ditetapkan. Hanya satu tipe yang boleh ditetapkan.	Tambahan

Sumber: (<https://developers.google.com/places/web-service/search>, 2018)

2.6.3 Contoh Permintaan Tempat

Contoh berikut adalah permintaan pencarian untuk tempat dengan jenis *groceries* dalam radius 500 m dari Gedung Informatika Untan, Pontianak, yang berisi kata *market* dalam nama tempat yang dicari. Pada penelitian ini lokasi untuk pencarian akan diambil dengan menggunakan *Current Place* pada *Places SDK* untuk dikirimkan pada permintaan terdekat.

```
https://maps.googleapis.com/maps/api/place/nearbysearch/json?
location=-0.055623,109.348594
&radius=500
&type=groceries
&keyword=market
&key=API_KEY
```

2.7 Google Places SDK For Android

Places SDK untuk Android memungkinkan pengembang aplikasi untuk membangun *location-aware apps* yang merespon secara kontekstual terhadap bisnis lokal dan berbagai tempat lain yang ada disekitar perangkat. Beberapa fitur yang disediakan oleh *Places SDK* untuk android yaitu *Places* yang menyediakan

akses ke basis data Google tentang tempat-tempat lokal dan informasi bisnis serta lokasi perangkat sekarang melalui kode program dan *Autocomplete* yang menyediakan *widgets* untuk menampilkan prediksi tempat sebagai respon dari kueri pencarian oleh user (Google Developers, n.d.).

Place Picker adalah suatu fitur yang memungkinkan pengguna bisa melihat tempat-tempat yang ada disekitar lokasi pengguna. Cara kerja *place picker* yaitu dengan menggunakan interface UI yang sudah disediakan oleh Google, kemudian menggunakan *intent* untuk menjalankan fungsi *place picker*. Ketika dijalankan, *place picker* akan mendeteksi koordinat lokasi pengguna dan kemudian menampilkan lokasi-lokasi lain yang berada disekitar pengguna. Setelah itu pengguna dapat memilih satu lokasi tertentu dan mengambil data lokasi yang dipilih tersebut. Pada penelitian ini *place picker* digunakan pada aplikasi untuk memudahkan pengguna aplikasi memasukkan informasi tempat belanja ketika membuat catatan belanja.

Place autocomplete digunakan untuk pencarian toko yang menampilkan prediksi tempat berdasarkan kata kunci yang diberikan oleh pengguna. Pada penelitian ini juga menggunakan fitur *current place* pada aplikasi yang bekerja dengan cara mengambil daftar tempat yang kemungkinan merupakan lokasi pengguna untuk kemudian diambil data *latitude* dan *longitude* dari tempat pada urutan pertama yang ada pada daftar tempat. *Latitude* dan *longitude* kemudian digunakan sebagai parameter untuk mendapatkan toko terdekat.

Lokasi yang didapatkan dari *current place* merupakan kemungkinan tempat berupa tempat bisnis atau *points of interest*. Berikut merupakan kode program untuk mendapatkan daftar tempat kemungkinan lokasi perangkat dapat dilihat pada kode program 2.4.

Kode Program 2.4 Kode program *current place*

```
1. Task<PlaceLikelihoodBufferResponse> placeResult =
    mPlaceDetectionClient.getCurrentPlace(null);
```

2.8 RESTful Web Service

REST atau *Representational State Transfer* merupakan salah satu jenis *web service* yang pertama kali diperkenalkan pada tahun 2000 oleh Roy Fielding (T. Fielding, 2000). REST merupakan standar dalam arsitektur web yang menggunakan

protocol HTTP untuk pertukaran data (Richardson & Ruby, 2007). REST banyak digunakan pada *web service* yang berorientasi pada *resource*. Orientasi pada *resource* atau sumber daya yang dimaksud adalah orientasi yang menyediakan sumber daya sebagai layanannya, bukan kumpulan-kumpulan dari aktifitas yang mengolah sumber daya tersebut. Cara kerja REST yaitu server menyediakan jalur untuk akses *resource* atau data, sedangkan pada sisi client melakukan akses *resource* untuk digunakan atau ditampilkan. Setiap sumber data diidentifikasi menggunakan link URI. Data atau *resource* yang dihasilkan umumnya dalam format JSON atau XML, sedangkan REST API pada penelitian ini *resource* yang dihasilkan dalam format JSON. *Web service* dengan menggunakan arsitektur REST baik untuk digunakan sebagai *back-end* dari aplikasi berbasis *mobile* karena cara akses yang lebih mudah dan hasil data yang dikirimkan berformat JSON sehingga ukuran *file* menjadi lebih kecil.

Berikut adalah metode HTTP yang umumnya digunakan dalam arsitektur REST :

1. GET untuk menyediakan akses yang digunakan untuk membaca sumber data.
2. POST untuk membuat data baru.
3. PUT untuk memperbarui data yang tersedia.
4. DELETE untuk menghapus data.

Web service adalah salah satu bentuk sistem perangkat lunak yang didesain untuk mendukung interaksi mesin ke mesin melalui jaringan (World Wide Web Consortium, 2004). Sistem *web service* memungkinkan sebuah fungsi yang ada di dalam *web service* dapat digunakan oleh aplikasi lain tanpa perlu mengetahui detail pemrograman yang terdapat didalamnya. Contoh implementasi dari *web service* antara lain adalah SOAP dan REST. Sistem yang menggunakan arsitektur dari REST dapat disebut dengan RESTful *Web service* (Su & Chiang, 2012). Layanan ini menggunakan metode HTTP untuk menerapkan konsep arsitektur REST.

Cara kerja dari RESTful *web service* yaitu *client* mengirimkan sebuah data atau *request* melalui HTTP *request* dan kemudian server merespon melalui HTTP *response*. Komponen dari HTTP *request* yaitu :

1. HTTP *method* yang digunakan misalnya GET, POST, PUT, DELETE.

2. *Uniform Resource Identifier* (URI) untuk mengidentifikasi lokasi *resource* pada server.
3. *HTTP Version*, menunjukkan versi dari HTTP yang digunakan, contoh HTTP v1.1.
4. *Request Header*, berisi metadata untuk HTTP *request* seperti *type client/browser*, format yang didukung oleh client, format dari *body* pesan.
5. *Request body*, berisi konten dari data.

Sedangkan komponen dari HTTP *response* yaitu :

1. *Status/Response Code*, mengindikasikan status server terhadap *resource* yang diminta.
2. *Response Header*, berisi metadata untuk HTTP *Response* seperti tipe server, panjang konten, tipe konten, dan waktu response.
3. *Response Body*, berisi konten dari data yang diberikan.

2.9 PHP

PHP Hypertext Preprocessor (PHP) merupakan salah satu yang banyak digunakan sebagai *server-side scripting language* untuk mengembangkan aplikasi berbasis *web*. PHP adalah *script* yang digunakan untuk membuat halaman *website* yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh *client*. Mekanisme ini menyebabkan informasi yang diterima *client* selalu baru atau *up to date*. Semua *script PHP* dieksekusi pada server dimana *script* tersebut dijalankan (Anhar, 2010).

2.10 JSON (JavaScript Object Nation)

JSON (*JavaScript Object Nation*) adalah sebuah standar yang berbasis teks yang didesain untuk pertukaran data yang mudah dibaca manusia (Sriparasa, 2013). Kegunaan dari JSON :

1. Digunakan saat menulis aplikasi berbasis *JavaScript* yang memerlukan ekstensi *browser* dan *website*.
2. Format JSON digunakan untuk mentransmisikan data yang berstruktur melalui jaringan.
3. Biasanya digunakan untuk mentransmisikan data antara server dan aplikasi *web*.

4. *Web Service* dan *API* menggunakan format *JSON* untuk menyediakan data publik.
5. Dapat digunakan pada bahasa pemrograman modern.

2.11 MySQL

MySQL adalah *system management database* relasional. Suatu relasional data dalam tabel terpisah. Hal ini memungkinkan kecepatan dan fleksibilitas. Tabel-tabel yang dihubungkan dengan relasi yang ditentukan membuatnya bisa mengkombinasikan data dari beberapa tabel dari suatu permintaan. Bagian *SQL* dari kata *MySQL* berasal dari *Structured Query Language* yaitu bahasa yang paling umum digunakan untuk mengakses *database*. Konektivitas, kecepatan, dan keamanannya membuat *MySQL* cocok untuk pengaksesan *database* pada internet (Anhar, 2010).

2.11.1 Perintah-perintah MySQL

Query pada MySQL adalah *query* yang diciptakan dengan menggunakan pernyataan-pernyataan SQL. Pernyataan (*statement*) SQL dapat digolongkan atas tiga golongan, yaitu:

1. *Data Definition Language* (DDL) yang mendefinisikan struktur data. Perintah-perintah SQL yang termasuk DDL antara lain *create*, *alter*, dan *drop*.
2. *Data Manipulation Language* (DML) yang mencari (*query*) dan mengubah (*modify*) suatu tabel. Perintah-perintah SQL yang termasuk DML antara lain *select*, *insert*, *update*, dan *delete*.
3. *Data Control Language* (DCL) yang mengatur hak-hak (*privilege*) untuk seorang pemakai *database*. Perintah-perintah SQL yang termasuk DCL antara lain *grant* dan *revoke*.

2.12 Unified Modelling Language (UML)

Untuk membantu dalam pengembangan perangkat lunak dikenal istilah pemodelan. Salah satu pemodelan yang saat ini paling banyak digunakan oleh pengembang perangkat lunak adalah UML (*Unified Modelling Language*). UML adalah standar bahasa yang sering digunakan dalam bidang industri untuk mendefinisikan *requirement* (kebutuhan), membuat analisis dan desain, serta

menggambarkan arsitektur dalam pemrograman berorientasi objek.

UML terbagi atas 3 (tiga) kategori, yaitu diagram struktur (*structure diagram*), diagram kelakuan sistem (*behaviour diagram*), dan diagram interaksi (*interaction diagram*) (A.S & Shalahuddin, 2014).

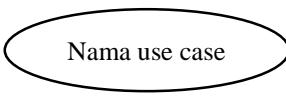
1. *Structure diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. *Structure diagram* terdiri dari *class diagram*, *object diagram*, *component diagram*, *composite diagram*, *package diagram*, dan *deployment diagram*.
2. *Behaviour diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. *Behaviour diagram* terdiri dari *use case diagram*, *activity diagram*, *state machine system*.
3. *Interaction diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. *Interaction diagram* terdiri dari *sequence diagram*, *communication diagram*, *timing diagram*, *interaction overview diagram*.

2.12.1 Use Case Diagram

Use case diagram merupakan pemodelan untuk kelakuan (*behaviour*) sistem yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi itu (A.S & Shalahuddin, 2014).

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

Tabel 2.2 Simbol-simbol *Use Case Diagram*

No	Simbol	Deskripsi
1.	<i>Use case</i> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawali fase nama <i>use case</i> .
2.	Aktor/ <i>actor</i>	Orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat di luar sistem

No	Simbol	Deskripsi
		yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3.	Asosiasi/association 	Komunikasi antara aktor dan <i>use case</i> yang berpatisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor.
4.	Menggunakan / include <i>/ uses</i> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini.
5.	Generalisasi/generalization 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah <i>use case</i> dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :
		<p>Arah panah mengarah pada <i>use case</i> yang menjadi generalisasinya (umum).</p>
6.	Ekstensi/extend 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> dimana <i>use case</i> yang ditambahkan dapat berdiri sendiri walaupun tanpa <i>use case</i> tambahan itu, mirip dengan prinsip <i>inheritance</i> pada pemrograman berorientasi objek, biasanya <i>use case</i> tambahan memiliki nama depan yang sama dengan <i>use case</i>

No	Simbol	Deskripsi
		<p>yang ditambahkan, misal</p> <p>Arah panah mengarah pada <i>use case</i> yang ditambahkan, biasanya <i>use case</i> yang menjadi <i>extend</i>-nya merupakan jenis yang sama dengan <i>use case</i> yang menjadi induknya.</p>

Sumber : (A. S & Shalahuddin, 2014, p. 156)

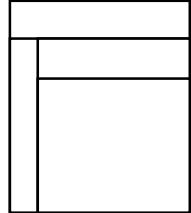
2.12.2 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Perlu diperhatikan bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem (A.S & Shalahuddin, 2014).

Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

Tabel 2.3 Simbol-simbol *Activity Diagram*

No	Simbol	Deskripsi
1.	Status awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan/ <i>decision</i>	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.

No	Simbol	Deskripsi
4.	Penggabungan/join 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.	Status akhir 	Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

Sumber : (A. S & Shalahuddin, 2014).

2.12.3 Class Diagram

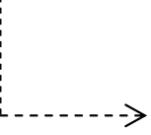
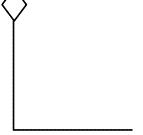
Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan *method* atau operasi (A.S & Shalahuddin, 2014). Berikut penjelasan atribut dan *method* :

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau *method* adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Berikut adalah simbol-simbol yang ada pada diagram kelas :

Tabel 2.4 Simbol-simbol *Class Diagram*

No	Simbol	Deskripsi
1.	Kelas 	Kelas pada struktur sistem.
2.	Antarmuka/interface 	Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek.

No	Simbol	Deskripsi
3.		Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
4.		Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
5.		Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).
6.		Relasi antar kelas dengan makna kebergantungan antar kelas.
7.		Relasi antar kelas dengan makna semua bagian (<i>whole-part</i>).

Sumber : (A. S & Shalahuddin, 2014).

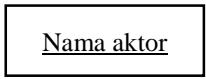
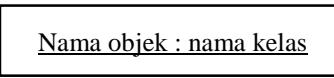
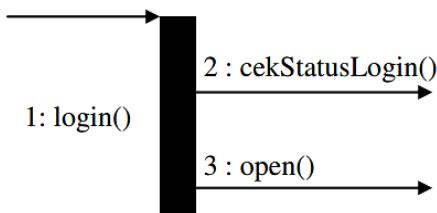
2.12.4 Sequence Diagram

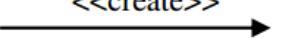
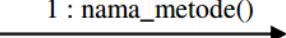
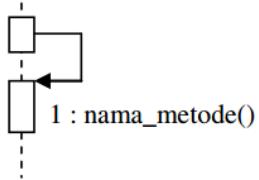
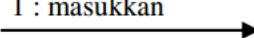
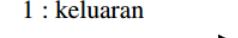
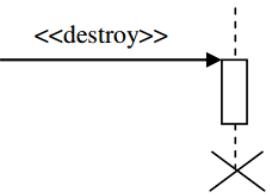
Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dengan *message* yang dikirimkan dan diterima antar objek. Menggambarkan diagram sekuen harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya diagram sekuen yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses

sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup dalam diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak (A.S & Shalahuddin, 2014).

Berikut adalah simbol-simbol yang ada pada diagram sekuen :

Tabel 2.5 Simbol-simbol Sequence Diagram

No	Simbol	Deskripsi
1.	Aktor  Atau  Tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat diluar sistem yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar, tapi aktor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda diawali frase nama aktor.
2.	Garis hidup/lifeline 	Menyatakan kehidupan suatu objek
3.	Objek 	Menyatakan objek yang berinteraksi pesan
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  Maka cekStatusLogin() dan open() dilakukan didalam metode login(). Aktor tidak memiliki

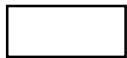
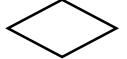
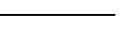
No	Simbol	Deskripsi
		waktu aktif.
5.	Pesan tipe <i>create</i> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe <i>call</i> 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri, 
		Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
7.	Pesan tipe <i>send</i> 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
8.	Pesan tipe <i>return</i> 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9.	Pesan tipe <i>destroy</i> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaliknya jika ada <i>create</i> maka ada <i>destroy</i> .

Sumber : (A. S & Shalahuddin, 2014).

2.13 Entity Relationship Diagram (ERD)

Entity Relationship Diagram atau ERD adalah suatu model jaringan yang menggambarkan *layout* (susunan) penyimpanan data dari sebuah sistem ER-Diagram yang menggambarkan data-data dalam keadaan diam (data yang disimpan) (Jogiyanto, 2005).

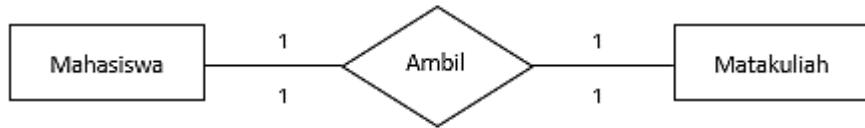
Komponen - komponen ERD:

1. *Entity* adalah segala sesuatu yang dapat dijelaskan dengan data kelompok benda atau obyek diberi nama dengan kata benda. Simbol dari *entity* adalah seperti ini 
2. *Relationship* adalah suatu assosiasi antar satu atau beberapa *entity*, diberi nama dengan kata benda. Simbol dari *relationship* adalah 
3. *Atribute* adalah properti atau karakteristik suatu *entity relationship*. Simbol dari atribut adalah 
4. Kardinalitas (*Cardinality*) menunjukkan jumlah maksimum entitas yang dapat berelasi dengan entitas pada entitas yang lain. Hubungan yang terjadi dapat berbentuk: hubungan satu ke satu (1:1), hubungan satu kebanyak (1:M), atau banyak ke satu (M:1), dan hubungan banyak ke banyak (N:M). simbol dari kardinalitas adalah 

Berikut ini adalah menggambarkan hubungan dari *key* suatu entitas memiliki kegiatan lain pada entity tersebut.

1. *One to One* (satu ke satu)

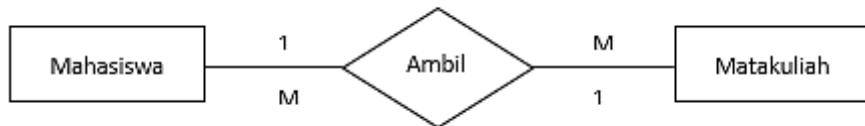
One to One adalah hubungan yang menggambarkan bahwa *key* suatu entitas hanya memiliki satu atribut yang berhubungan dengan satu atribut yang lain pada *entity* tersebut. Relasi di bawah ini menggambarkan bahwa untuk setiap entitas A (mahasiswa) berpasangan dengan maksimal 1 entitas di himpunan entitas B (matakuliah). Asumsinya yaitu 1 orang mahasiswa hanya dapat mengambil 1 matakuliah, begitu juga jika dibalik. 1 matakuliah hanya dapat diambil oleh 1 mahasiswa. Oleh karena itu relasi ini berkardinalitas *One to One* (satu ke satu).



Gambar 2.5 Relasi *One to One*

2. *One to Many* (satu ke banyak)

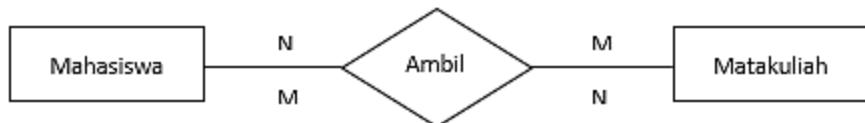
One to Many adalah hubungan yang menggambarkan bahwa *key* suatu entitas memiliki banyak kegiatan entitas lain. Relasi di bawah ini menggambarkan bahwa untuk setiap entitas di himpunan entitas A (*mahasiswa*) berpasangan dengan banyak entitas di himpunan B (*matakuliah*). Asumsinya yaitu 1 mahasiswa dapat mengambil banyak matakuliah. Oleh karena itu relasi ini berkardinalitas *One to Many* (satu ke banyak)



Gambar 2.6 Relasi *One to Many*

3. *Many to Many* (banyak ke banyak)

Many to Many adalah hubungan yang menggambarkan bahwa *key* suatu entitas memiliki banyak kegiatan entitas lain. Relasi di bawah menggambarkan bahwa untuk setiap entitas di himpunan entitas A (*mahasiswa*) berpasangan dengan banyak entitas di himpunan B (*matakuliah*). Asumsinya yaitu untuk 1 matakuliah dapat diambil oleh banyak mahasiswa



Gambar 2.7 Relasi *Many to Many*

2.14 Pengujian Perangkat Lunak

Dalam penelitian ini dilakukan beberapa pengujian, berikut adalah pengujian yang dilakukan.

2.14.1 Pengujian *Black Box*

Ada beberapa metode yang dapat digunakan untuk memilih data pengujian, salah satunya adalah metode *black box*. Metode *Black-Box* yaitu data pengujian dipilih berdasarkan spesifikasi masalah tanpa memperhatikan detail internal dari program (A.S, 2009), untuk memeriksa apakah program dapat berjalan dengan benar.

2.14.1.1 *Sample Testing*

Pengujian *sample testing* merupakan salah metode pengujian *black box* yang dilakukan dengan menggunakan beberapa nilai yang terpilih, kemudian mengintegrasikan nilai pada kasus uji (A.S, 2009). Pada penelitian ini, pengujian *sample testing* digunakan untuk menguji fitur rekomendasi harga terendah pada aplikasi menghasilkan harga terendah dengan baik. Pengujian ini dilakukan dengan mengambil beberapa data yang ada pada basis data, kemudian dilakukan pengecekan hasil pada aplikasi sudah sesuai dengan kriteria atau belum sesuai.

2.14.1.2 *Bitbar Testing*

Bitbar *Testing* dibuat oleh Testdroid *technology*, yang menyediakan *automation testing* dan *manual testing* menggunakan perangkat Android dan iOS asli. Sehingga pengembang perangkat lunak dapat membuat aplikasi, permainan, dan *website* berkualitas tinggi.

Bitbar *Testing* menyediakan alat pengujian berbasis *cloud* yang fleksibel. Menggunakan bitbar *testing* pengembang perangkat lunak dapat melakukan otomasi pengujian aplikasi dengan berbagai *test framework* populer seperti *espresso*, *appium*, dan *calabash* atau menguji secara manual untuk melihat bagaimana pengguna menggunakan aplikasi dengan menggunakan *remote device control*.

Pengujian dengan menggunakan bitbar *testing* mempunyai kelebihan konkurensi tidak terbatas sehingga dapat melakukan pengujian perangkat dengan berbagai jumlah perangkat, dari sepuluh sampai seribu, dan secara paralel menggunakan perangkat dalam jumlah besar yang tersedia *online*.

Hasil pengujian didapatkan secara cepat dengan semua data, *logs*, *videos*, *screenshots*, serta grafik *CPU* dan *memory usage* hasil pengujian, sehingga

pengembang dapat mendeteksi dan menyelesaikan permasalahan lebih cepat dan sesuai waktu untuk menghasilkan aplikasi yang lebih baik. Pada penelitian ini Bitbar *Testing* digunakan sebagai pengujian aplikasi dari segi fungsionalitas, kompatibilitas, dan performa. Metode ini lebih efektif dan mendalam karena pengujian ini mencakup pengujian *blackbox* (fungsionalitas), ditambah dengan pengujian kompatibilitas, dan performa yang dilakukan secara cepat dan dalam beberapa perangkat sekaligus.

2.15 Kajian Terkait

Go Frendi Gunawan (2013), Program Studi Teknik Informatika, Sekolah Tinggi Informatika dan Komputer Indonesia dalam jurnal yang berjudul Otomatisasi Sistem Rekomendasi Layanan Kesehatan untuk Berobat Berbasis WebGIS menjelaskan bahwa tujuan dari penelitiannya yaitu menciptakan Aplikasi WebGIS Kesehatan yang mampu memberikan rekomendasi sarana kesehatan di Kota Malang secara lengkap. Penelitian ini menggunakan Google Maps sebagai salah satu pilihan layer peta induk untuk menampilkan hasil rekomendasi. Data rekomendasi tempat berobat dari hasil proses otomatisasi pemilihan layanan kesehatan berdasarkan *request* jenis penyakit yang diderita *client*. Hasil rekomendasi tempat berobat dapat diurutkan berdasarkan jarak terdekat dengan batasan jarak maksimal 8 km yang dapat diubah sesuai input dari *client*.

I Ketut Resika Arthana (2018), Program Studi Pendidikan Teknik Informatika, Universitas Pendidikan Ganesha dalam jurnal yang berjudul Prototype Aplikasi Mobile Preservasi Warisan Budaya Indonesia Berbasis Crowdsourcing menjelaskan bahwa tujuan dari penelitiannya yaitu mengembangkan prototype Aplikasi Warisan Budaya (WADAYA) Indonesia dengan mengadopsi konsep *crowdsourcing* dimana informasi warisan budaya berasal dari partisipasi masyarakat umum dengan menambahkan warisan budaya baru yang belum tercatat pada sistem atau menambahkan informasi pada warisan budaya yang sudah ada pada sistem. Pada penelitian ini, *geotagging* sesuai koordinat posisi yang dideteksi oleh GPS pada telepon seluler digunakan untuk memperoleh lokasi pada peta yang dimanfaatkan untuk menentukan lokasi objek budaya yang ada.

Kajian terkait dapat dilihat pada tabel 2.6

Tabel 2.6 Tabel Perbandingan Penelitian

No	Penulis	Judul	Keterangan
1.	Go Frendi Gunawan (2013), Program Studi Teknik Informatika, Sekolah Tinggi Informatika dan Komputer Indonesia	Otomatisasi Sistem Rekomendasi Layanan Kesehatan untuk Berobat Berbasis WebGIS	<ul style="list-style-type: none"> - Rekomendasi berobat ini berdasarkan dari penyakit yang diderita masyarakat. - Input berupa penyakit pasien dan lokasi pasien - Output berupa rekomendasi tempat berobat dengan ranking yang bisa diubah urutannya, misalnya berdasarkan kedekatan lokasi, relevansi dengan spesialisasi penanganan penyakit.
2	I Ketut Resika Arthana (2018), Program Studi Pendidikan Teknik Informatika, Universitas Pendidikan Ganesha	Prototype Aplikasi Mobile Preservasi Warisan Budaya Indonesia Berbasis Crowdsourcing	<ul style="list-style-type: none"> - Aplikasi dikembangkan dengan konsep <i>crowdsourcing</i> dimana informasi warisan budaya berasal dari masyarakat. - Aplikasi digunakan untuk memberikan informasi objek budaya. - Masyarakat dapat mencari objek budaya berdasarkan jenis, nama, maupun lokasi terdekat.

Tabel 2.7 Tabel Penelitian yang Akan Dilakukan

No	Penulis	Judul	Keterangan
1.	Fany Ayu Anggreany (2019), Universitas Tanjungpura Pontianak	Aplikasi Rekomendasi Layanan Harga Produk Terendah dengan Partisipasi Masyarakat Berdasarkan Catatan Belanja	<ul style="list-style-type: none"> - Sistem menggunakan partisipasi masyarakat untuk menghimpun informasi harga produk. - Partisipasi masyarakat dilakukan pada aplikasi dengan membuat catatan belanja berdasarkan struk belanja. - Menggunakan Google Maps untuk menambahkan informasi tempat belanja. - Data catatan belanja meliputi nama, harga, jumlah, dan tempat produk dibeli sesuai struk belanja. - Rekomendasi dihasilkan berdasarkan total harga produk terendah dan toko dengan jarak terdekat.

BAB III

METODOLOGI PENELITIAN

3.1 Alat dan Bahan Penelitian

3.1.1 Alat Penelitian

Alat penelitian yang digunakan dalam penelitian ini adalah:

- a. Diagram alir sistem, digunakan untuk menggambarkan bagaimana sistem dapat menghasilkan rekomendasi harga terendah untuk pengguna.
- b. *Unified Modelling Language* (UML). UML digunakan untuk menggambarkan batasan sistem dan fungsi-fungsi sistem secara umum.

3.1.1.1 Perangkat Keras

Perangkat keras yang digunakan untuk merancang aplikasi dalam penelitian ini adalah:

- a. Satu unit laptop Asus X450CC, dengan spesifikasi Intel Core i3-3217U @1.80GHz, 6.00 GB RAM DDR3, Harddisk 500 GB
- b. Smartphone Xiaomi Redmi Note 5 dengan spesifikasi Qualcomm Snapdragon 636 octa-core, 32 GB internal memory, 3 GB RAM, GPS, Android versi 8.1.0.

3.1.1.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam penelitian ini adalah:

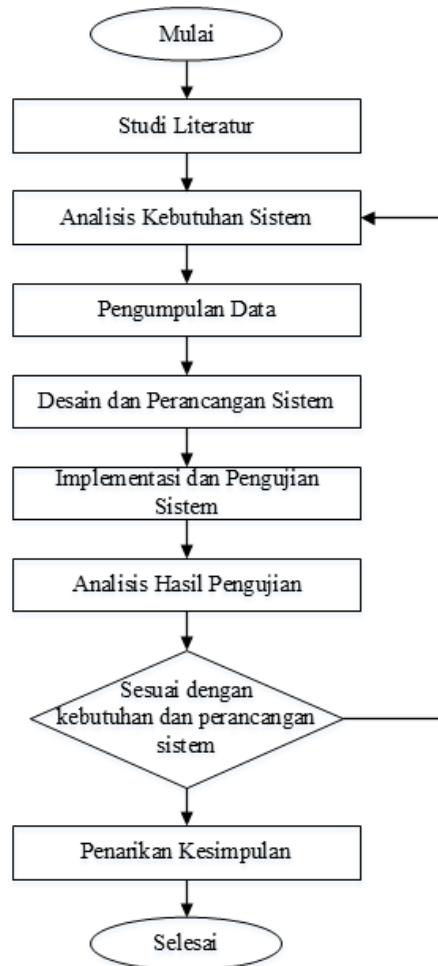
- a. Sistem Operasi Windows 10.
- b. Android Studio 3.2.1 sebagai aplikasi untuk pengembangan aplikasi android.
- c. Sublime Text 3 sebagai aplikasi untuk text editor.
- d. PHP version 7.1.1 sebagai bahasa pemrograman yang digunakan.
- e. MySQL 5.14.1 sebagai pengelola sistem database.
- f. SQLite sebagai penyimpanan data lokal pada aplikasi android.
- g. Codeigniter version 3.1.9 sebagai Framework bahasa pemrograman php.
- h. Postman version 6.5.3 sebagai aplikasi untuk menguji coba *web service*.
- i. Astah Professional 8.1.0 sebagai alat bantu pembuatan diagram.
- j. Moqups sebagai alat bantu pembuatan perancangan aktivitas *layout*.

3.1.2 Bahan Penelitian

Bahan penelitian berupa data *barcode* dan nama produk kebutuhan sehari-hari seperti sabun cuci, sabun mandi, makanan, dan minuman. Data tersebut akan digunakan pada aplikasi untuk memudahkan pengguna melakukan input data produk catatan belanja. Selain itu data struk belanja juga digunakan sebagai bahan penelitian untuk kemudian diproses menjadi rekomendasi harga produk terendah.

3.2 Metode Penelitian

Metode penelitian merupakan beberapa tahapan yang dirancang dan dijadikan sebagai panduan dalam melakukan penelitian. Tahapan pada penelitian yang akan dilakukan digambarkan pada diagram alir penelitian pada Gambar 3.1 sebagai berikut.



Gambar 3.1 Diagram Alir Penelitian

Berdasarkan pada Gambar 3.1 dapat dijelaskan diagram alir penelitian sebagai berikut.

3.2.1 Studi Literatur

Studi literatur dilakukan dengan mencari referensi ilmu pengetahuan yang berhubungan dengan pembuatan sistem seperti referensi tentang catatan belanja, aplikasi sejenis yang juga menerapkan partisipasi masyarakat, android studio, JSON, UML (*Unified Modeling Language*) maupun kritik dan saran melalui penelitian-penelitian sebelumnya.

3.2.2 Analisis Kebutuhan

Sistem dibangun berbasis android dan *website* dengan data pada android yang tersimpan secara lokal dan juga tersimpan pada server agar aplikasi tetap dapat berjalan secara *offline*. Pada penggunaan aplikasi secara *offline* yang dapat dilakukan pengguna aplikasi hanya mencatat belanjaan tanpa menginput tempat belanja, menghitung pengeluaran, serta membuat rencana belanja. Pembatasan penggunaan aplikasi secara *offline* disebabkan dalam memilih tempat belanja data tempat tidak disimpan secara lokal sehingga diperlukan koneksi internet untuk dapat mengakses data tempat. Pada saat menambahkan item belanja, data produk catatan belanja dapat *diinput* oleh pengguna aplikasi dengan cara *scan barcode* produk atau dengan mengetik nama produk. Pengguna dapat melihat daftar harga produk mulai dari harga terendah dari berbagai toko yang berbeda-beda. Melalui pencarian produk pengguna dapat menambahkan produk tersebut ke rencana belanjanya.

Pada aplikasi web pengguna yaitu admin dapat menambahkan data produk, mengubah data produk, dan menghapus data produk yang telah ditambahkan oleh pengguna melalui aplikasi android.

3.2.3 Pengumpulan Data

Pengumpulan data dilakukan untuk memperoleh data-data yang diperlukan oleh sistem yang akan dibuat seperti data produk meliputi *barcode* dan nama produk serta data struk belanja. Data produk dikumpulkan dari website Sinarmart.com, Smarco.co.id, Supermart.co.id, Suzuya.id, idmarco.com, dan klikbigmart.com. Kemudian data produk berupa nama dan *barcode* dimasukkan ke dalam *database* melalui web admin yang digunakan sebagai pilihan produk untuk menginput catatan belanja.

Struk belanja yang digunakan merupakan struk belanja dari minimarket dan

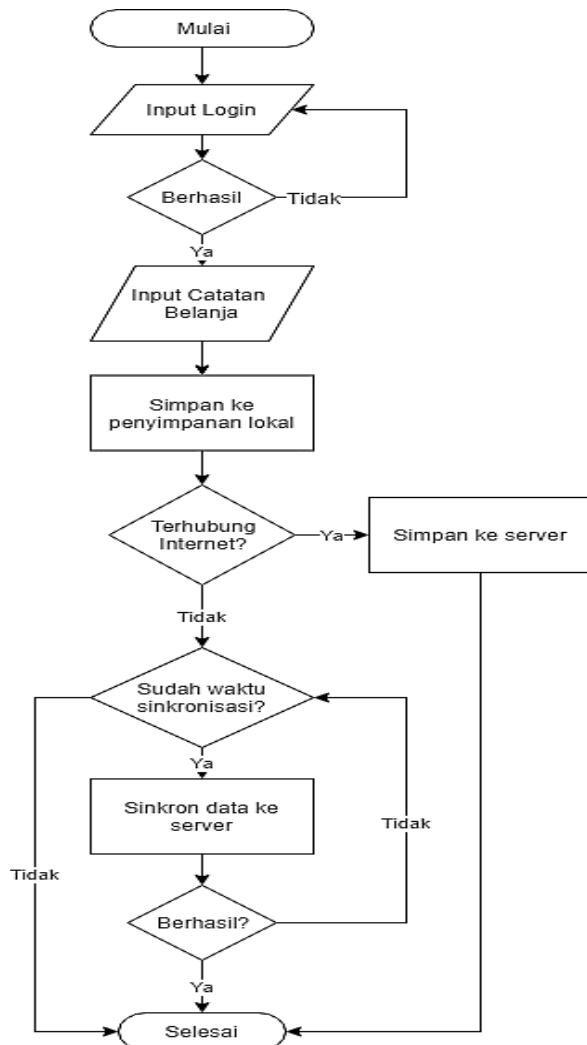
supermarket di Kota Pontianak. Produk pada struk belanja yang dimasukkan ke dalam sistem hanya produk yang memiliki *barcode*. Kemudian data produk meliputi nama produk, jumlah pembelian produk, dan harga satuan produk dari struk belanja diinput melalui aplikasi.

3.2.4 Perancangan Sistem

Dalam melakukan perancangan sistem terdapat tahapan-tahapan yang dilakukan yaitu:

3.2.4.1 Diagram Alir

Diagram alir sistem merupakan diagram yang menggambarkan sistem yang akan dibangun. Adapun diagram alir sistem yang akan dibangun dapat dilihat pada Gambar 3.2 berikut:



Gambar 3.2 Diagram Alir Sistem

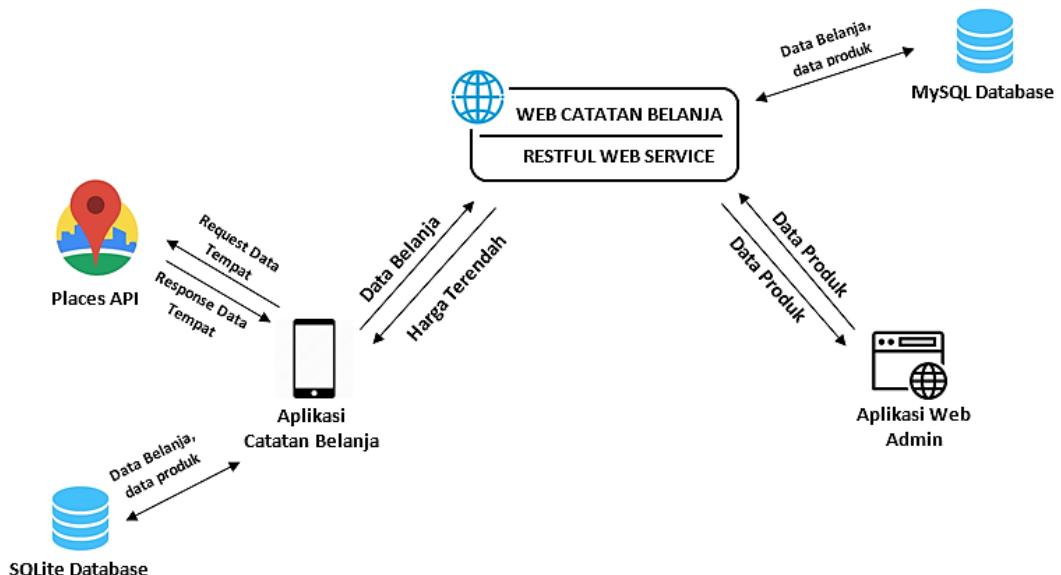
Data catatan belanja yang berasal dari struk belanja akan diinputkan oleh

pengguna melalui aplikasi. Kemudian data catatan belanja akan disimpan pada penyimpanan lokal, jika aplikasi terhubung internet maka data akan disimpan ke server. Data yang tersimpan pada penyimpanan lokal akan dilakukan sinkronisasi berkala setiap 10 jam sekali dengan menggunakan *scheduler*.

Data catatan belanja yang berasal dari struk belanja akan diinputkan oleh pengguna melalui aplikasi dan disimpan ke server. Tidak semua data belanja yang diinputkan oleh *user* disimpan sebagai riwayat harga. Data yang tersimpan pada server kemudian akan dilakukan pengecekan tanggal berbelanja, tempat berbelanja, nama produk, dan harga produk pada database. Jika terdapat tanggal, tempat, nama, dan harga produk yang sama maka data tersebut tidak akan disimpan sebagai riwayat harga produk. Setelah data tersimpan sebagai riwayat harga, untuk mendapatkan harga produk terendah, *user* diharuskan melakukan pencarian produk. Pada pencarian produk, riwayat harga produk di *database* disaring berdasarkan nama produk dan kemudian diurutkan dari harga yang terendah sampai harga tertinggi.

3.2.4.2 Arsitektur Sistem

Adapun arsitektur sistem yang akan dibangun dapat dilihat pada Gambar 3.3.



Gambar 3.3 Arsitektur Sistem

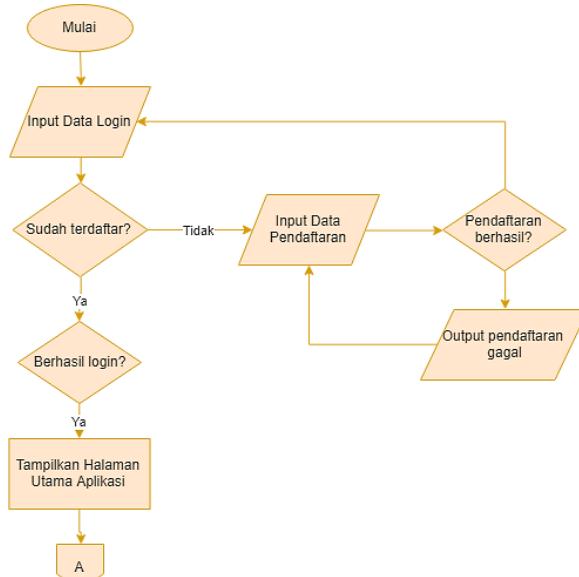
Berdasarkan Gambar 3.3 proses yang terjadi yaitu masyarakat yang menggunakan aplikasi catatan belanja akan membuat catatan hasil belanjaan dari

struk belanja yang diterima dari tempat belanja. Ketika menginputkan tempat belanja aplikasi akan melakukan request ke Places API dan akan memberikan response data tempat berupa JSON untuk ditampilkan pada maps atau ditampilkan dalam bentuk list yang dapat dipilih oleh pengguna. Semua item produk diinputkan beserta barcode produk, nama produk, dan harganya. Kemudian data catatan belanja tersebut disimpan ke dalam basis data SQLite yang berfungsi sebagai penyimpanan lokal. Kemudian jika terhubung ke internet, catatan belanja akan disimpan ke MySQL database pada server dan dapat diakses oleh pengguna lain melalui aplikasi dengan melakukan request ke server dan server akan memberikan response data harga terendah berupa JSON untuk kemudian ditampilkan oleh aplikasi. Jika banyak masyarakat yang melakukan pencatatan belanja pada aplikasi, maka akan terhimpun data produk beserta harganya. Dengan demikian konsep partisipasi masyarakat dapat menghasilkan data produk dengan informasi harga terbaru.

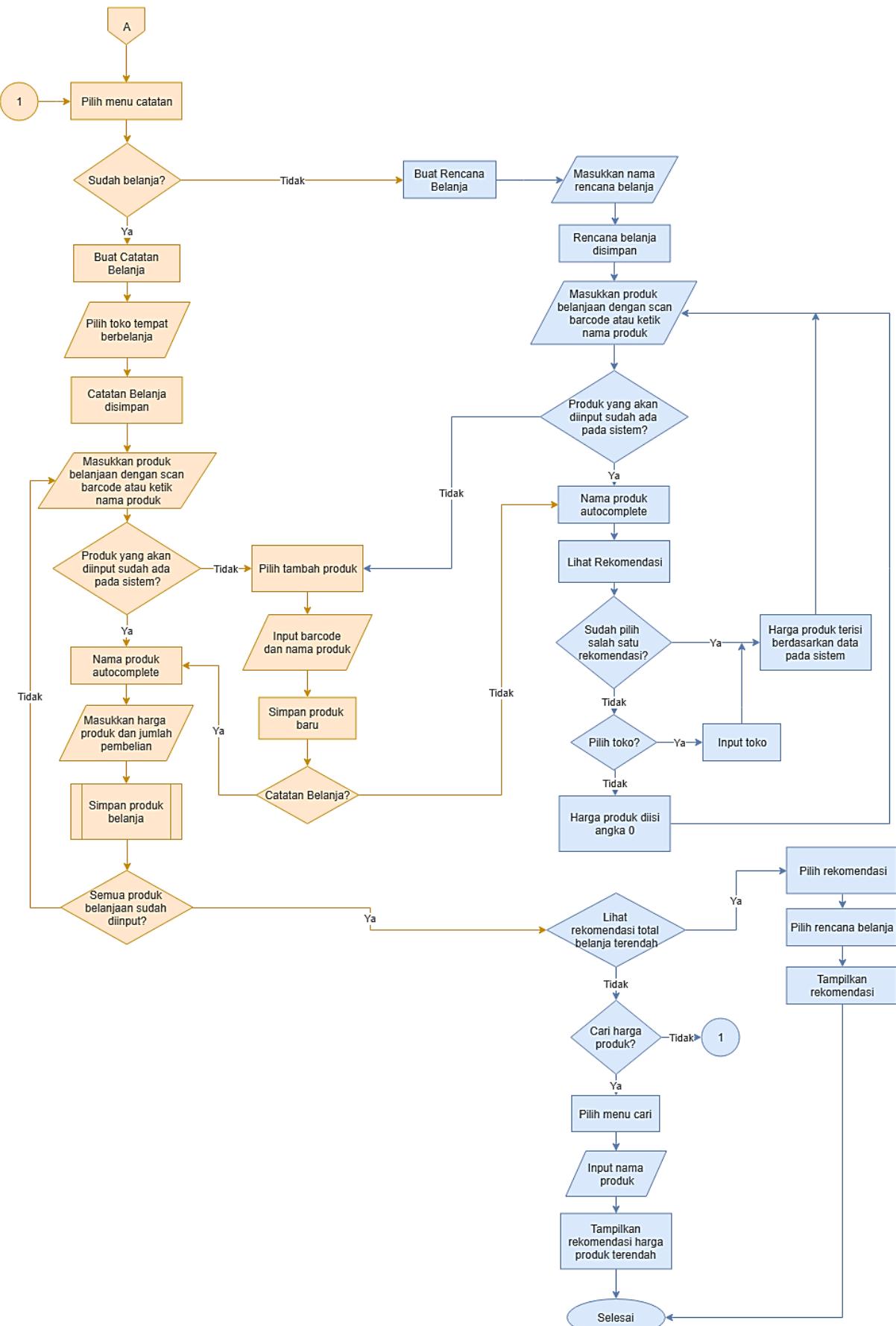
Web catatan belanja yang ditampilkan pada web browser merupakan sebuah admin panel yang digunakan sebagai pengelola data produk yang digunakan pada aplikasi untuk memudahkan pengguna dalam menambahkan item catatan belanja.

3.2.4.3 Proses Bisnis Sistem

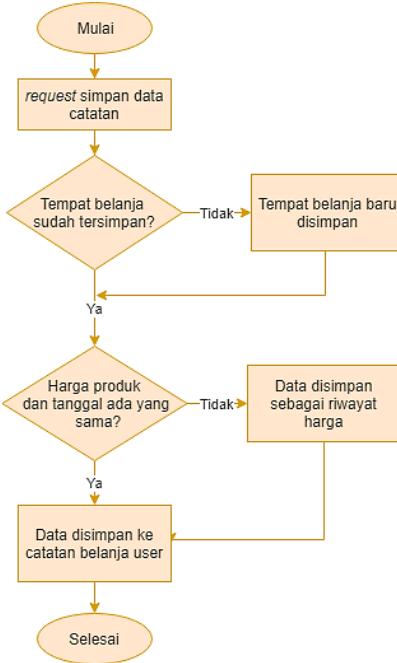
Proses bisnis merupakan kumpulan aktivitas yang menggambarkan alur bagaimana proses suatu bisnis dijalankan dan menghasilkan keluaran (*output*) yang bernilai bagi pelanggan atau *end-user*.



Gambar 3.4 Proses Bisnis Sistem



Gambar 3.5 Proses Bisnis Sistem



Gambar 3.6 Sub-process Simpan produk belanja

Berdasarkan proses bisnis pada gambar 3.4 dan 3.5, proses bisnis sistem dimulai dengan masyarakat sebagai pengguna aplikasi melakukan *login* aplikasi kemudian sistem akan mengecek apakah pengguna sudah terdaftar, jika belum maka pengguna akan melakukan pendaftaran terlebih dahulu dan kemudian melakukan *login*. Kemudian jika *login* berhasil aplikasi akan menampilkan halaman utama. Pada halaman utama, masyarakat sebagai pengguna aplikasi dapat memilih menu catatan dan membuat catatan belanja jika sudah belanja atau memiliki struk belanja yang akan dicatat. Pada catatan belanja ini masyarakat mencatat belanjaannya untuk dirinya sendiri dan juga berpartisipasi dalam memberikan informasi harga produk karena catatan belanja yang dimasukkan oleh pengguna akan disimpan pada server sebagai data harga produk dan data harga produk tersebut akan dapat dilihat oleh pengguna lainnya pada rekomendasi harga produk terendah.

Rekomendasi harga produk terendah dapat dilihat dengan membuat rencana belanja dan kemudian memasukkan produk yang akan dibeli. Setelah selesai memasukkan produk, pengguna dapat melihat rekomendasi berdasarkan total harga terendah dengan menekan tombol rekomendasi. Selain itu, pengguna juga dapat melihat rekomendasi harga satu produk dengan memilih menu cari dan menginputkan nama produk yang ingin dilihat harga terendahnya.

3.2.4.4 Perancangan Use Case Diagram

Diagram *use case* merupakan pemodelan untuk kelakuan (*behaviour*) sistem yang akan dibuat. Diagram ini digunakan untuk mendeskripsikan interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa yang sudah ada didalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

3.2.4.4.1 Definisi Aktor

Berikut adalah deskripsi pendefinisian aktor pada aplikasi rekomendasi harga produk terendah.

Tabel 3.1 Definisi Aktor

No.	Aktor	Deskripsi
1.	<i>User</i>	<i>User</i> merupakan orang yang memasukkan data catatan belanja. <i>User</i> akan dapat mengakses semua fitur yang ada aplikasi apabila sudah terdaftar pada aplikasi.
2.	Admin	Admin merupakan orang yang memasukkan dan mengelola data produk, dan juga memvalidasi data produk tambahan dari <i>User</i> .

3.2.4.4.2 Definisi Use Case

Berikut adalah deskripsi pendefinisian *use case* untuk android yang digunakan oleh *user* dan *use case* untuk web yang digunakan oleh admin pada sistem rekomendasi layanan harga produk terendah.

Tabel 3.2 Definisi Use Case User

No.	<i>Use case</i>	Deskripsi
UC01	Tambah catatan belanja	Merupakan proses menambahkan data catatan belanja.
UC02	Lihat rencana belanja	Merupakan proses menampilkan data rencana yang didalamnya <i>user</i> dapat melakukan proses menambah, mengubah, dan menghapus data rencana belanja.

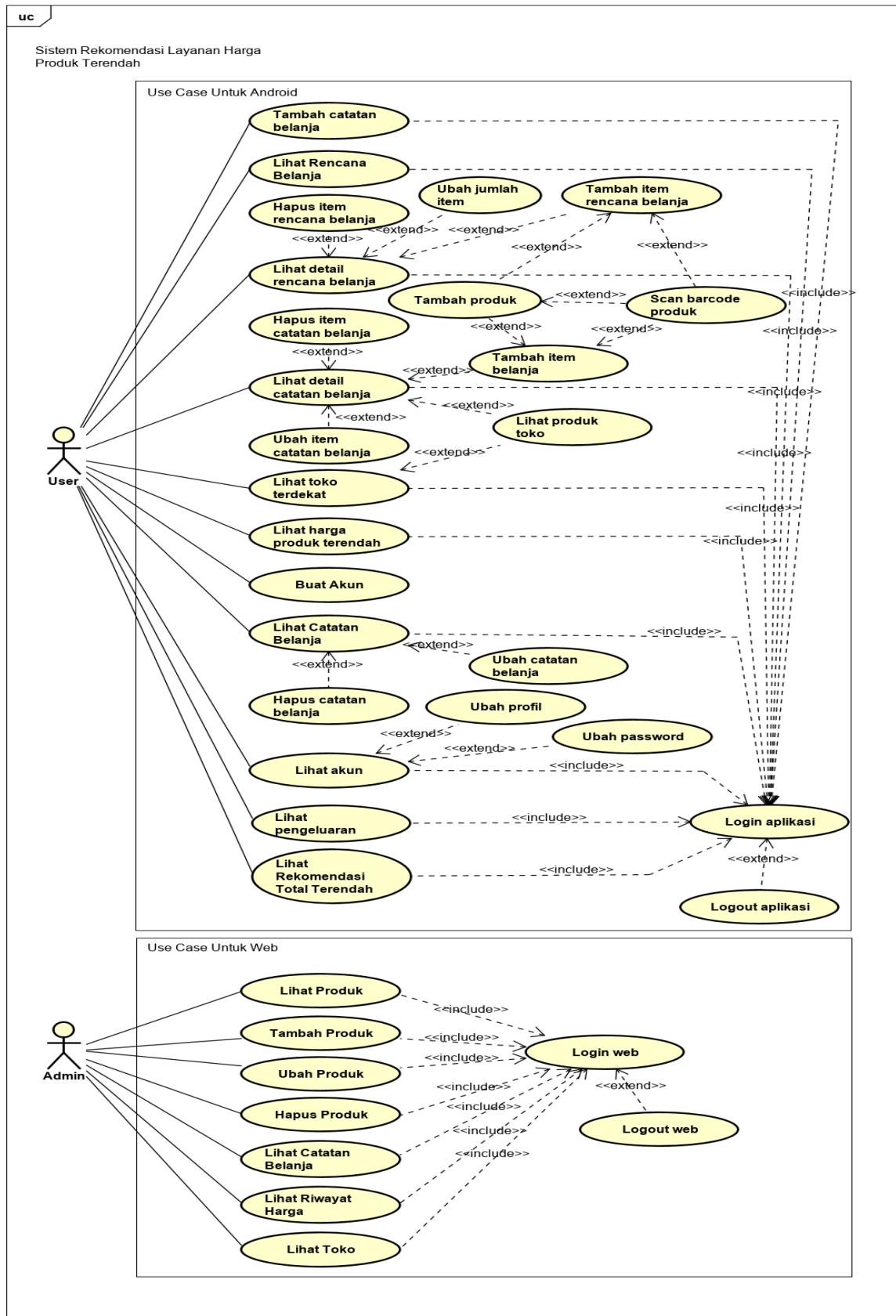
No.	Use case	Deskripsi
UC03	Lihat detail rencana belanja	Merupakan proses menampilkan detail rencana belanja yang didalamnya <i>user</i> dapat melakukan proses menambah dan menghapus data item rencana belanja, serta mengubah jumlah item.
UC04	Tambah produk	Merupakan proses menambahkan produk yang belum tersedia pada basis data.
UC05	Scan barcode produk	Merupakan proses scan barcode produk yang akan ditambahkan.
UC06	Lihat detail catatan belanja	Merupakan proses menampilkan detail catatan belanja yang didalamnya <i>user</i> dapat melakukan proses menambahkan, mengubah, dan menghapus item catatan belanja.
UC07	Lihat catatan belanja	Merupakan proses menampilkan data catatan belanja milik <i>user</i> yang didalamnya <i>user</i> dapat melakukan proses mengubah dan menghapus catatan belanja.
UC08	Lihat produk toko	Merupakan proses menampilkan data produk yang sudah tersimpan pada riwayat harga pada salah satu toko.
UC09	Lihat toko terdekat	Merupakan proses menampilkan toko terdekat dari lokasi <i>user</i> .
UC10	Lihat harga produk terendah	Merupakan proses menampilkan daftar harga produk mulai dari harga terendah ke harga tertinggi.
UC11	Buat akun	Merupakan proses membuat akun baru
UC12	Lihat akun	Merupakan proses melihat data akun milik <i>user</i> .
UC13	Ubah profil	Merupakan proses mengubah detail profil milik <i>user</i> .
UC14	Ubah password	Merupakan proses mengubah password milik

No.	Use case	Deskripsi
		<i>user.</i>
UC15	Lihat pengeluaran	Merupakan proses menampilkan total belanja berdasarkan catatan belanja <i>user</i> yang dapat ditampilkan berdasarkan total belanja perhari, perminggu, dan perbulan.
UC16	Lihat rekomendasi total terendah	Merupakan proses menampilkan rekomendasi total belanja terendah dari toko disekitar lokasi <i>user</i> berdasarkan rencana belanja <i>user</i> .
UC17	Login aplikasi	Merupakan proses untuk melakukan login <i>user</i> .
UC18	Logout aplikasi	Merupakan proses untuk melakukan logout <i>user</i> .

Tabel 3.3 Definisi Use Case Admin

No.	Use case	Deskripsi
UC19	Login web	Merupakan proses untuk melakukan login admin.
UC20	Tambah produk	Merupakan proses menambahkan data produk.
UC21	Ubah produk	Merupakan proses mengubah data produk.
UC22	Hapus produk	Merupakan proses menghapus data produk.
UC23	Lihat Catatan Belanja	Merupakan proses menampilkan semua catatan belanja yang ditambahkan oleh <i>user</i> aplikasi.
UC24	Lihat Riwayat Harga	Merupakan proses menampilkan semua riwayat harga yang dihasilkan dari catatan belanja <i>user</i> aplikasi.
UC25	Lihat Toko	Merupakan proses menampilkan semua toko dari catatan belanja <i>user</i> aplikasi.
UC26	Logout web	Merupakan proses untuk melakukan logout admin.

3.2.4.4.3 Diagram Use Case



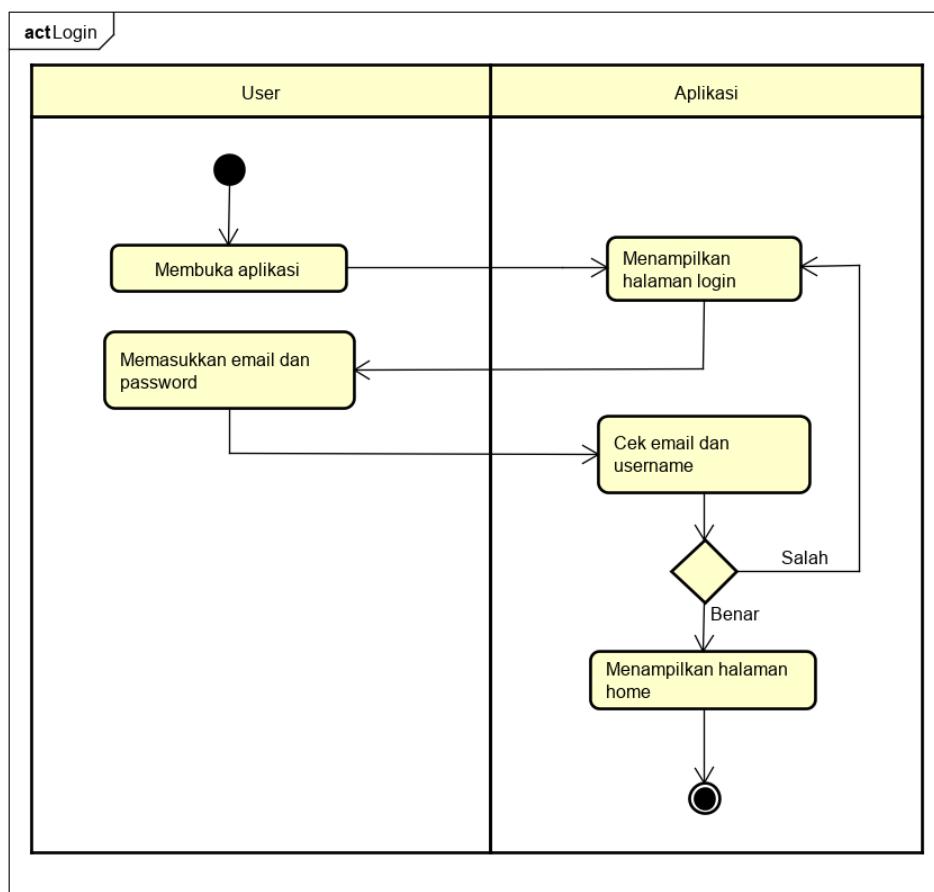
Gambar 3.7 Diagram *Use Case* Aplikasi

3.2.4.5 Perancangan Activity Diagram

Activity diagram menggambarkan alur kerja atau aktivitas pada sistem yang bertujuan untuk melihat alur proses sistem secara bertahap. *Activity diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case diagram*.

a. Activity Diagram Login User

Berikut adalah *activity diagram* user dapat dilihat pada Gambar 3.8.

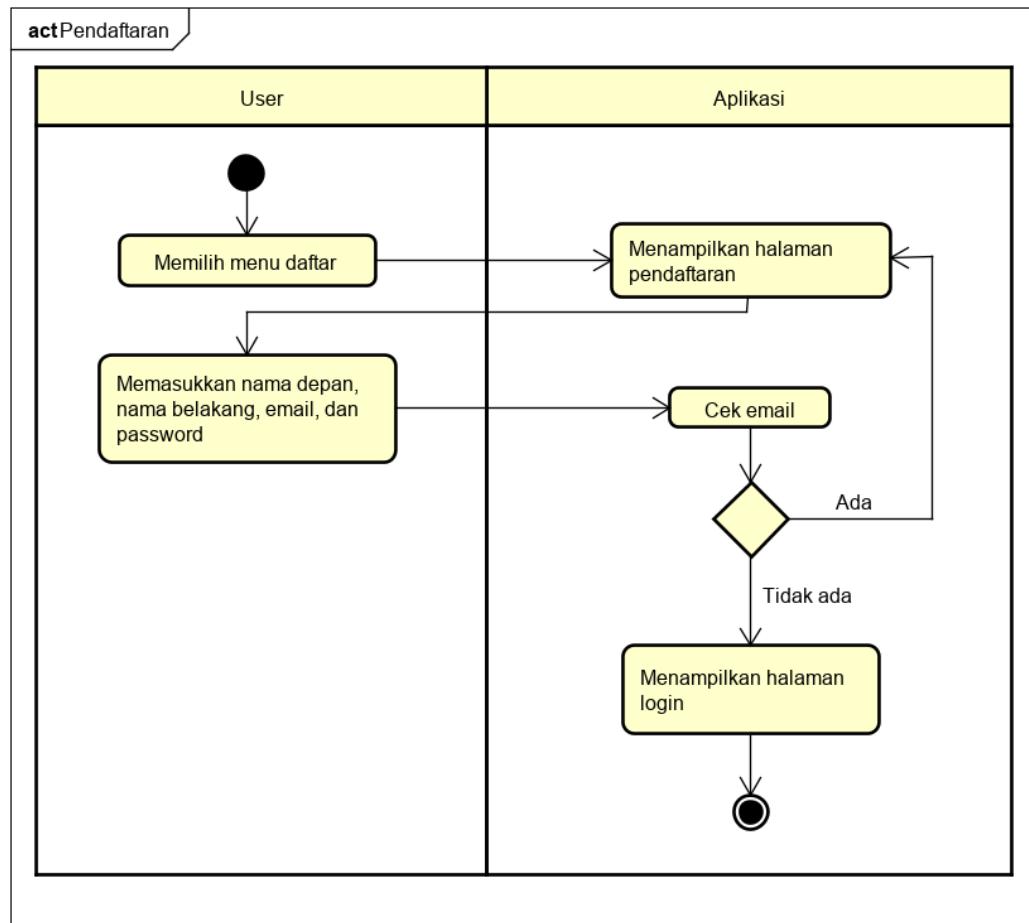


Gambar 3.8 Activity Diagram Login User

Pada Gambar 3.8, dapat dilihat bahwa saat login, user memasukkan email dan password, kemudian aplikasi akan mengecek jika email dan password benar maka aplikasi akan menampilkan halaman home, jika salah maka aplikasi akan kembali menampilkan halaman login.

b. Activity Diagram Buat Akun

Berikut adalah *activity diagram* buat akun dapat dilihat pada Gambar 3.9.

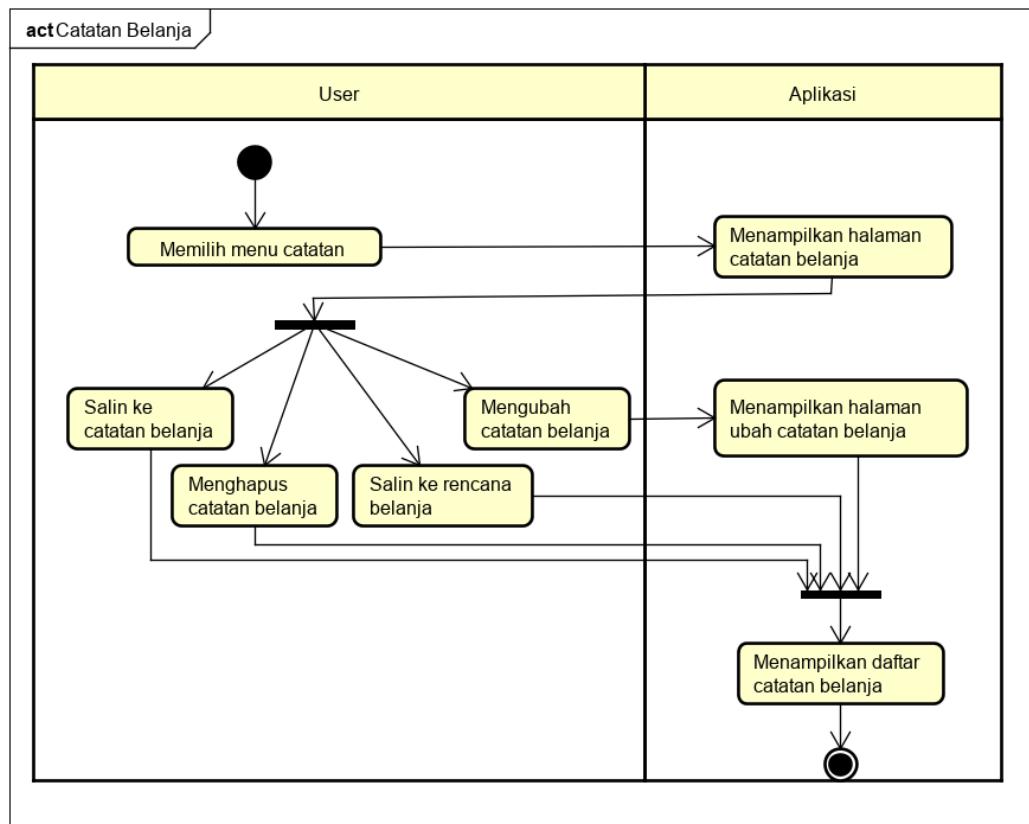


Gambar 3.9 Activity Diagram Buat Akun

Pada Gambar 3.9, dapat dilihat bahwa saat membuat akun, *user* memasukkan nama depan, nama belakang, email, dan *password*. Selanjutnya aplikasi mengecek email yang sudah terdaftar jika sudah ada maka akan kembali menampilkan halaman pendaftaran, jika email yang dimasukkan tidak ada maka pendaftaran berhasil dan aplikasi akan menampilkan halaman login.

c. Activity Diagram Catatan Belanja

Berikut adalah *activity diagram* catatan belanja dapat dilihat pada Gambar 3.10.

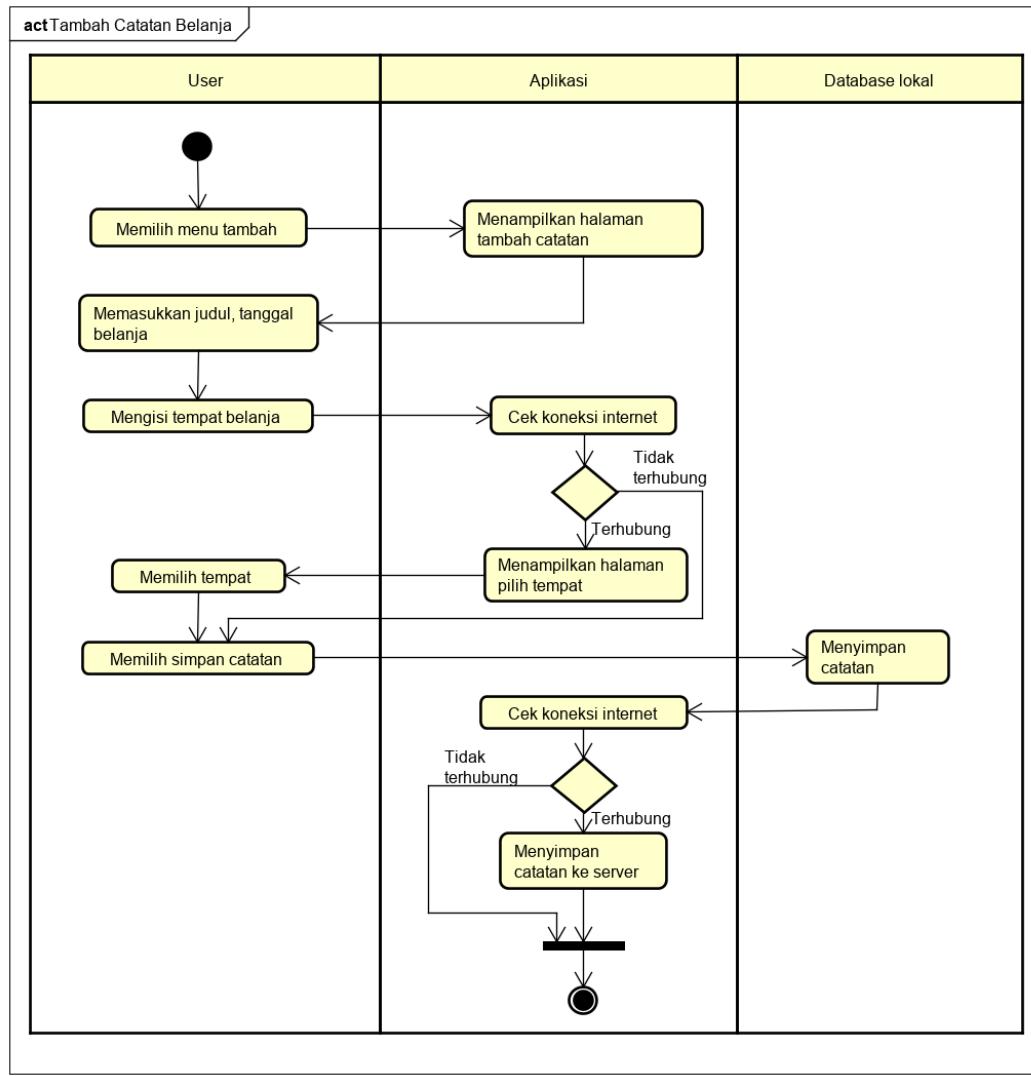


Gambar 3.10 *Activity Diagram* Catatan Belanja

Pada Gambar 3.10, dapat dilihat bahwa saat aplikasi menampilkan menu catatan belanja *user* dapat melakukan hapus dan ubah catatan belanja, jika *user* mengubah catatan belanja maka aplikasi akan menampilkan halaman ubah catatan belanja, setelah menampilkan halaman ubah catatan belanja aplikasi akan menampilkan daftar catatan belanja. Selanjutnya jika *user* menghapus catatan belanja maka aplikasi akan menampilkan daftar catatan belanja.

d. *Activity Diagram* Tambah Catatan Belanja

Berikut adalah *activity diagram* tambah catatan belanja dapat dilihat pada Gambar 3.11.

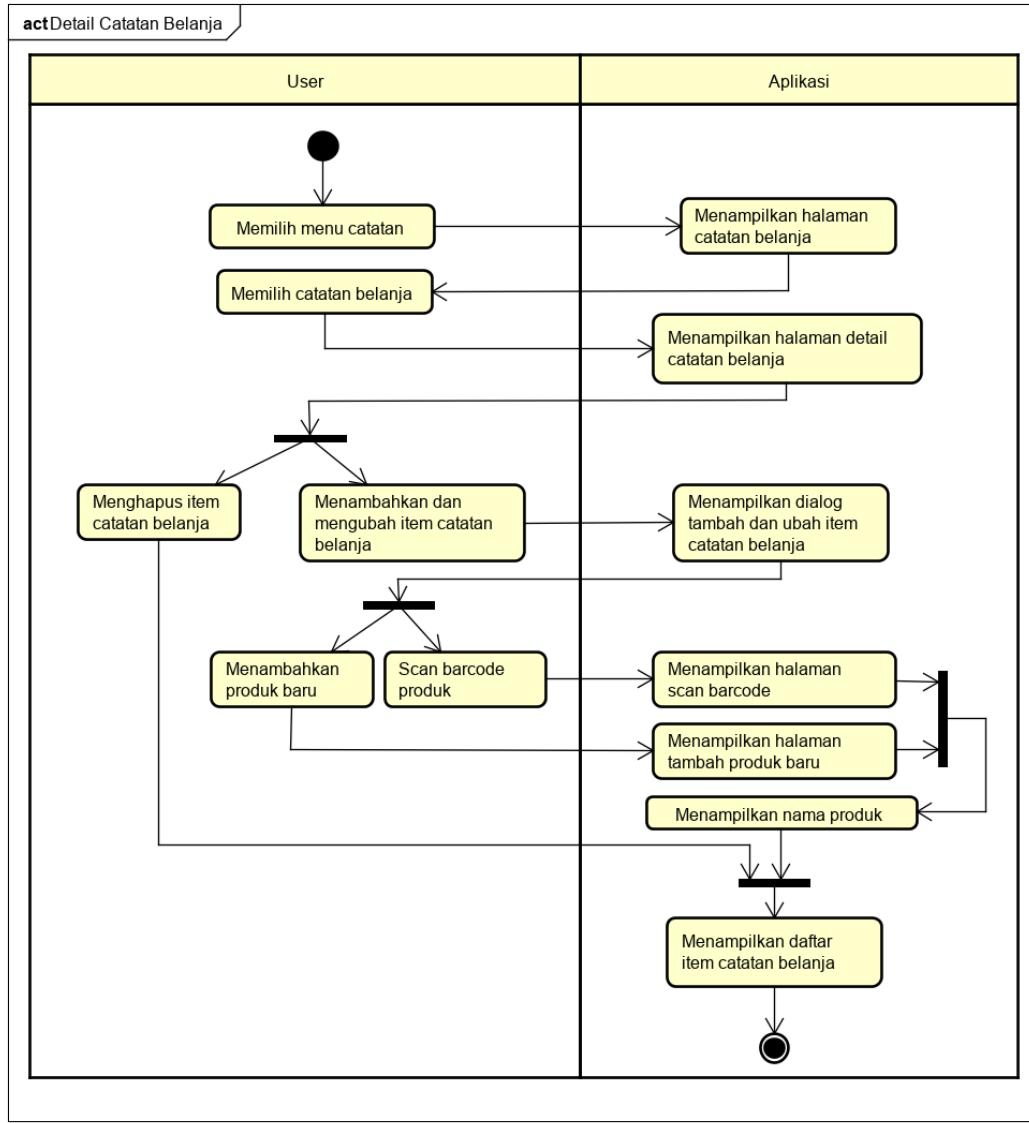


Gambar 3.11 Activity Diagram Tambah Catatan Belanja

Pada Gambar 3.11 dapat dilihat bahwa saat *user* memilih menu tambah maka aplikasi akan menampilkan halaman tambah catatan belanja, kemudian *user* memasukkan judul, dan tanggal belanja catatan belanja. Selanjutnya *user* mengisi tempat belanja dan aplikasi akan mengecek koneksi internet, jika tidak terhubung ke internet maka tidak perlu memilih tempat belanja dan kemudian dapat langsung menyimpan catatan, jika *user* terhubung ke internet maka aplikasi akan menampilkan halaman pilih tempat, kemudian *user* memilih tempat dan menyimpan catatan belanja. Selanjutnya catatan belanja akan langsung disimpan pada *database* lokal dan aplikasi akan mengecek koneksi internet, jika terhubung ke internet maka catatan belanja akan disimpan ke server.

e. *Activity Diagram* Detail Catatan Belanja

Berikut adalah *activity diagram* detail catatan belanja dapat dilihat pada Gambar 3.12.



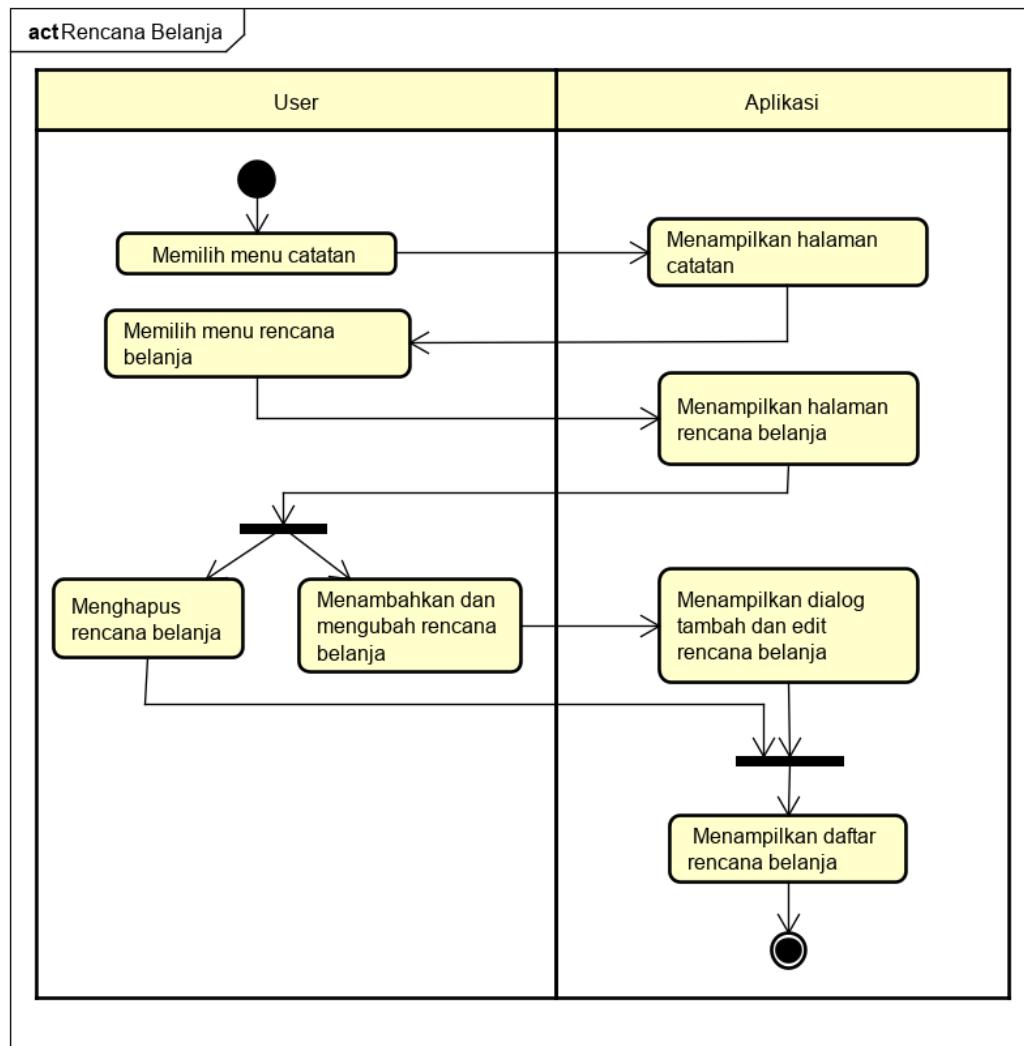
Gambar 3.12 *Activity Diagram* Detail Catatan Belanja

Pada Gambar 3.12, dapat dilihat bahwa *user* dapat mengakses halaman detail catatan belanja dengan memilih menu catatan dan kemudian memilih catatan belanja pada halaman detail catatan belanja. *User* dapat menambahkan dan mengubah item catatan kemudian aplikasi akan menampilkan dialog tambah dan ubah item catatan belanja. Pada dialog tambah dan ubah item, *user* dapat menambahkan produk baru jika produk yang akan diinput belum tersedia, *user* juga dapat menginputkan item catatan belanja dengan melakukan scan barcode dengan

memilih menu scan barcode produk kemudian aplikasi akan menampilkan halaman scan barcode, setelah proses scan barcode selesai maka aplikasi akan menampilkan nama produk yang sudah ditambahkan atau di scan pada dialog tambah dan ubah catatan belanja.

f. *Activity Diagram* Rencana Belanja

Berikut adalah *activity diagram* rencana belanja dapat dilihat pada Gambar 3.13.



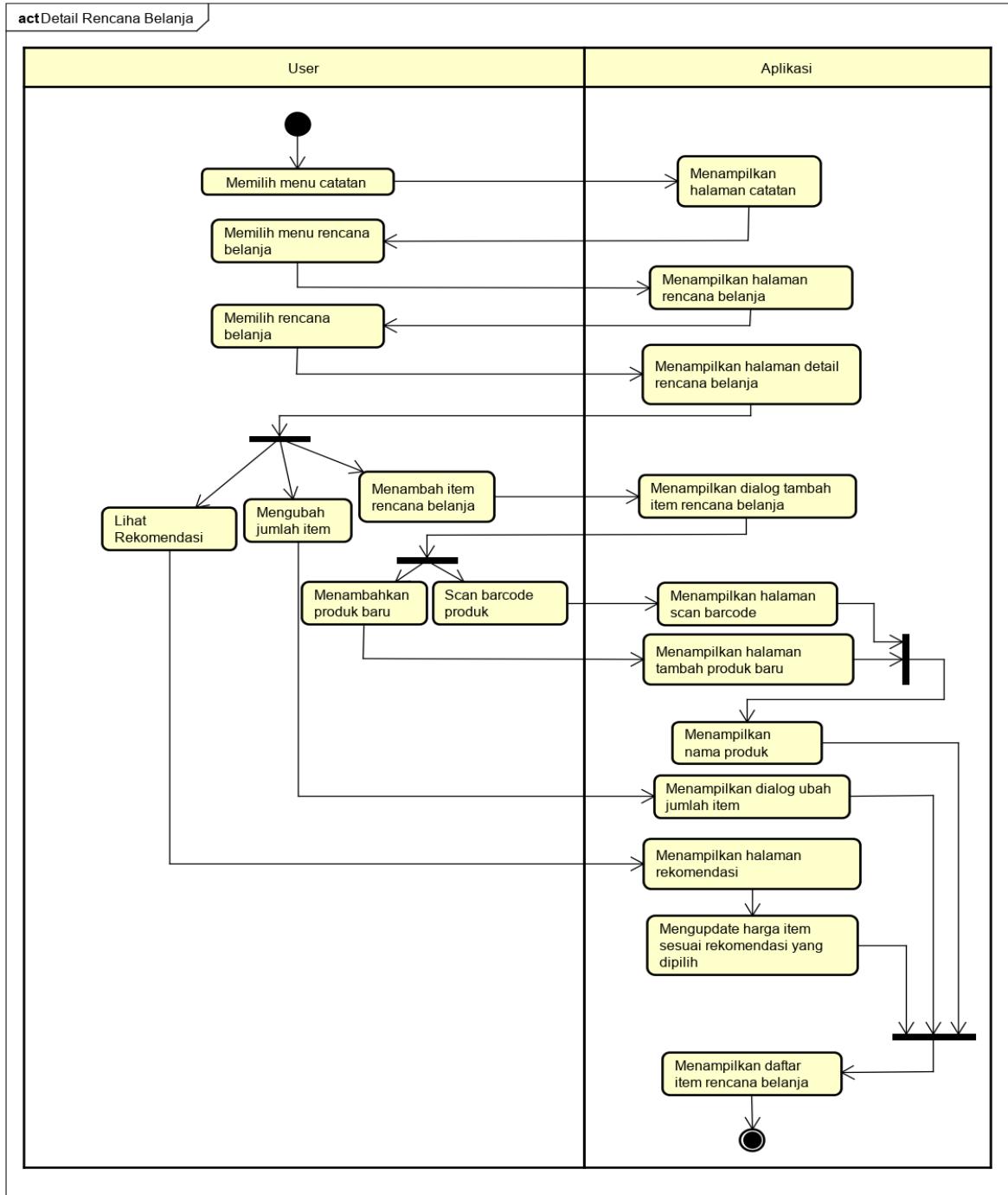
Gambar 3.13 *Activity Diagram* Rencana Belanja

Pada Gambar 3.13, dapat dilihat bahwa saat *user* memilih menu rencana belanja yang ada pada halaman catatan, aplikasi akan menampilkan halaman rencana belanja. Pada halaman rencana belanja *user* dapat melakukan hapus rencana belanja, menambahkan dan mengubah rencana belanja, jika *user*

menambahkan dan mengubah rencana belanja maka aplikasi akan menampilkan dialog tambah dan edit rencana belanja. Selanjutnya jika *user* memilih menghapus rencana belanja maka aplikasi akan menampilkan daftar rencana belanja.

g. Activity Diagram Detail Rencana Belanja

Berikut adalah *activity diagram* pengumuman dapat dilihat pada Gambar 3.14.

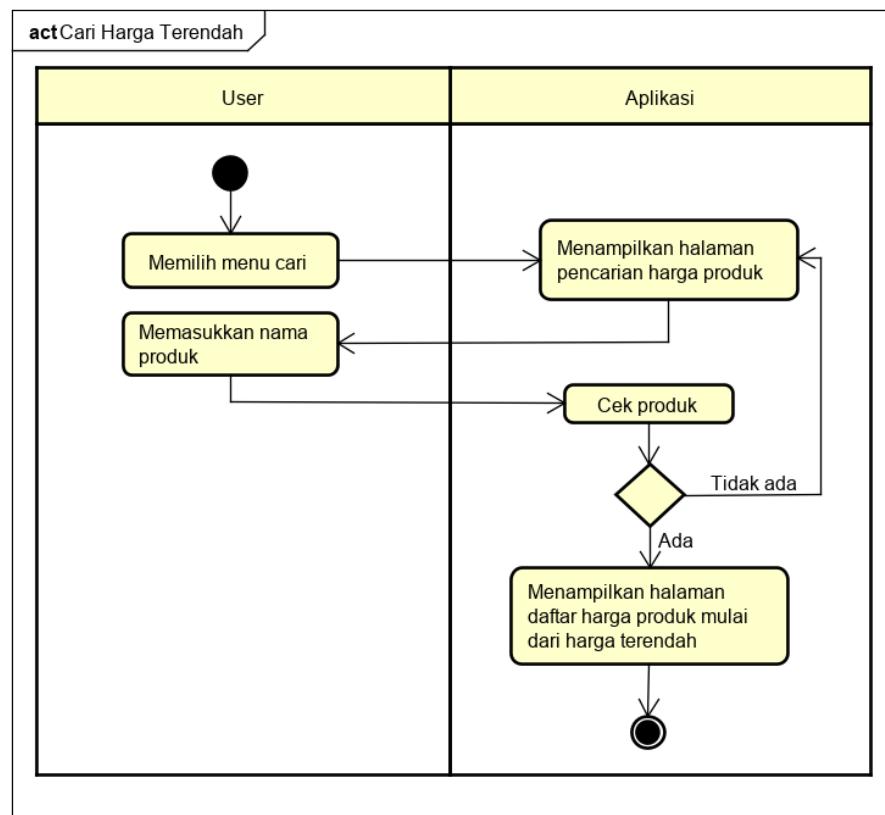


Gambar 3.14 Activity Diagram Detail Rencana Belanja

Pada Gambar 3.14, dapat dilihat bahwa *user* dapat melakukan lihat rekomendasi, ubah jumlah item, dan tambah item rencana belanja dengan cara memilih menu catatan kemudian pada halaman catatan *user* memilih menu rencana belanja. Pada saat *user* menambahkan item catatan belanja, aplikasi menampilkan dialog tambah item rencana belanja. Pada dialog tambah item rencana belanja, *user* dapat menambahkan item dengan *scan barcode* produk dan menambahkan produk baru setelah itu aplikasi akan menampilkan halaman *scan barcode* dan menampilkan halaman tambah produk baru. Pada saat *user* memilih melihat rekomendasi, aplikasi akan menampilkan hasil rekomendasi berdasarkan rencana belanja, kemudian memilih salah satu rekomendasi. Setelah *user* memilih salah satu rekomendasi, aplikasi akan mengupdate harga item yang sudah diinput dan kemudian menampilkan daftar item rencana belanja dengan harga item yang sudah terupdate.

h. Activity Diagram Cari Harga Terendah

Berikut adalah *activity diagram* cari harga terendah dapat dilihat pada Gambar 3.15.

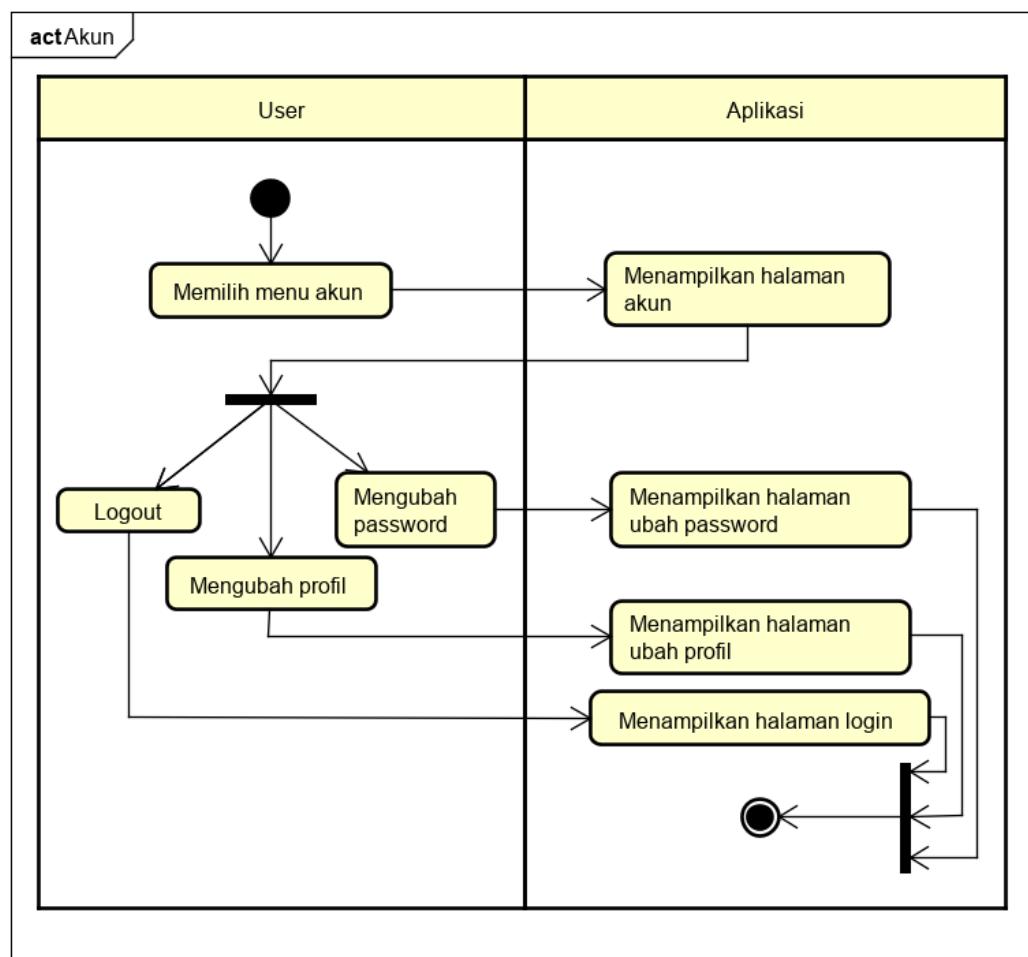


Gambar 3.15 Activity Diagram Cari Harga Terendah

Pada Gambar 3.15, dapat dilihat bahwa halaman pencarian harga produk dapat diakses oleh *user* dengan memilih menu cari dan kemudian memasukkan nama produk. Selanjutnya akan aplikasi akan mengecek produk berdasarkan nama produk, jika produk yang dicari tidak ada maka akan kembali ke halaman pencarian harga produk, jika produk ada maka daftar harga produk yang diurutkan mulai dari harga terendah akan ditampilkan.

i. Activity Diagram Akun

Berikut adalah *activity diagram* akun dapat dilihat pada Gambar 3.16.

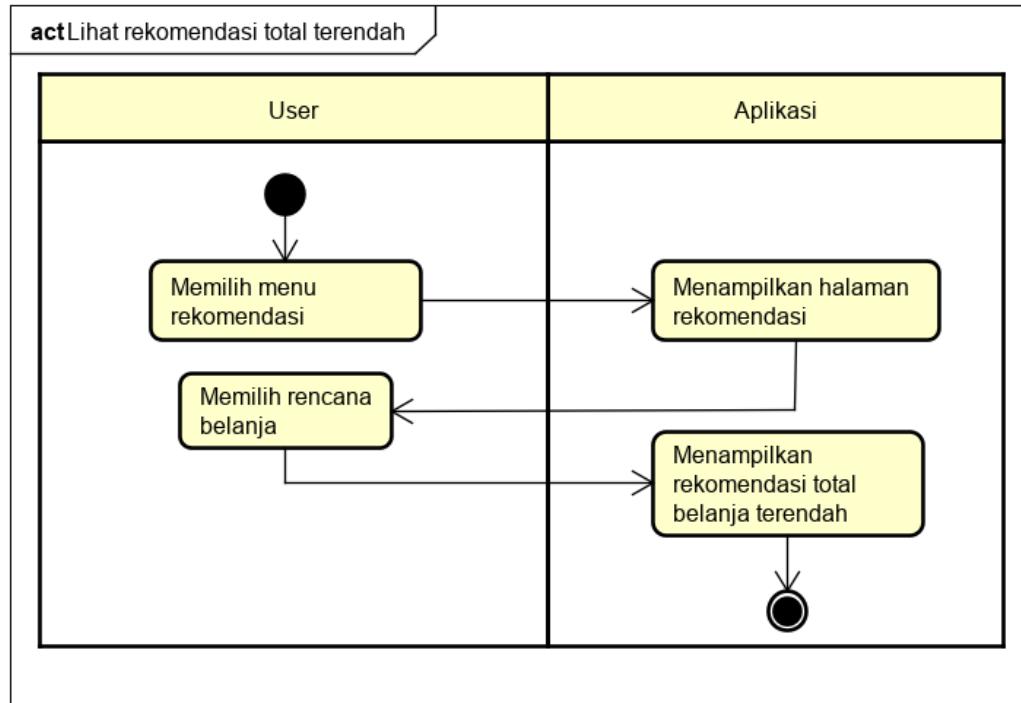


Gambar 3.16 *Activity Diagram* Akun

Pada Gambar 3.16, dapat dilihat bahwa pada halaman akun dapat dilakukan *logout*, ubah profil, dan ubah *password*. Pada saat mengubah *password*, ubah profil, dan *logout* maka ditampilkan halaman ubah password, halaman ubah profil, dan halaman *login*.

j. *Activity Diagram* Lihat Rekomendasi Total Terendah

Berikut adalah *activity diagram* lihat rekomendasi total terendah dapat dilihat pada Gambar 3.17.

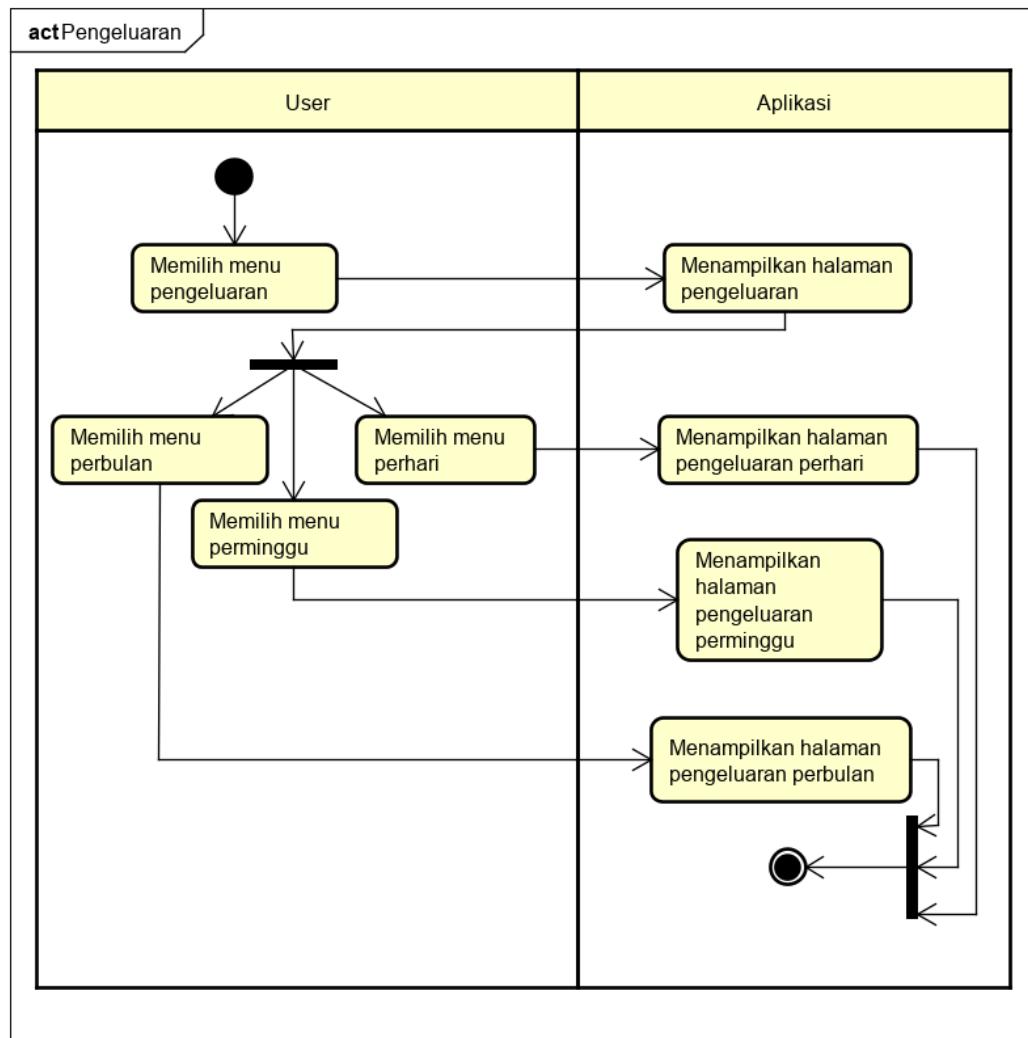


Gambar 3.17 *Activity Diagram* Lihat Rekomendasi Total Terendah

Pada Gambar 3.17, dapat dilihat bahwa dengan memilih menu rekomendasi aplikasi akan menampilkan halaman rekomendasi, kemudian *user* memilih salah satu rencana belanja. Setelah *user* memilih rencana belanja yang akan dilihat rekomendasinya, aplikasi akan menampilkan rekomendasi total belanja terendah dari berbagai toko disekitar lokasi *user*.

k. *Activity Diagram* Pengeluaran

Berikut adalah *activity diagram* pengeluaran dapat dilihat pada Gambar 3.18.

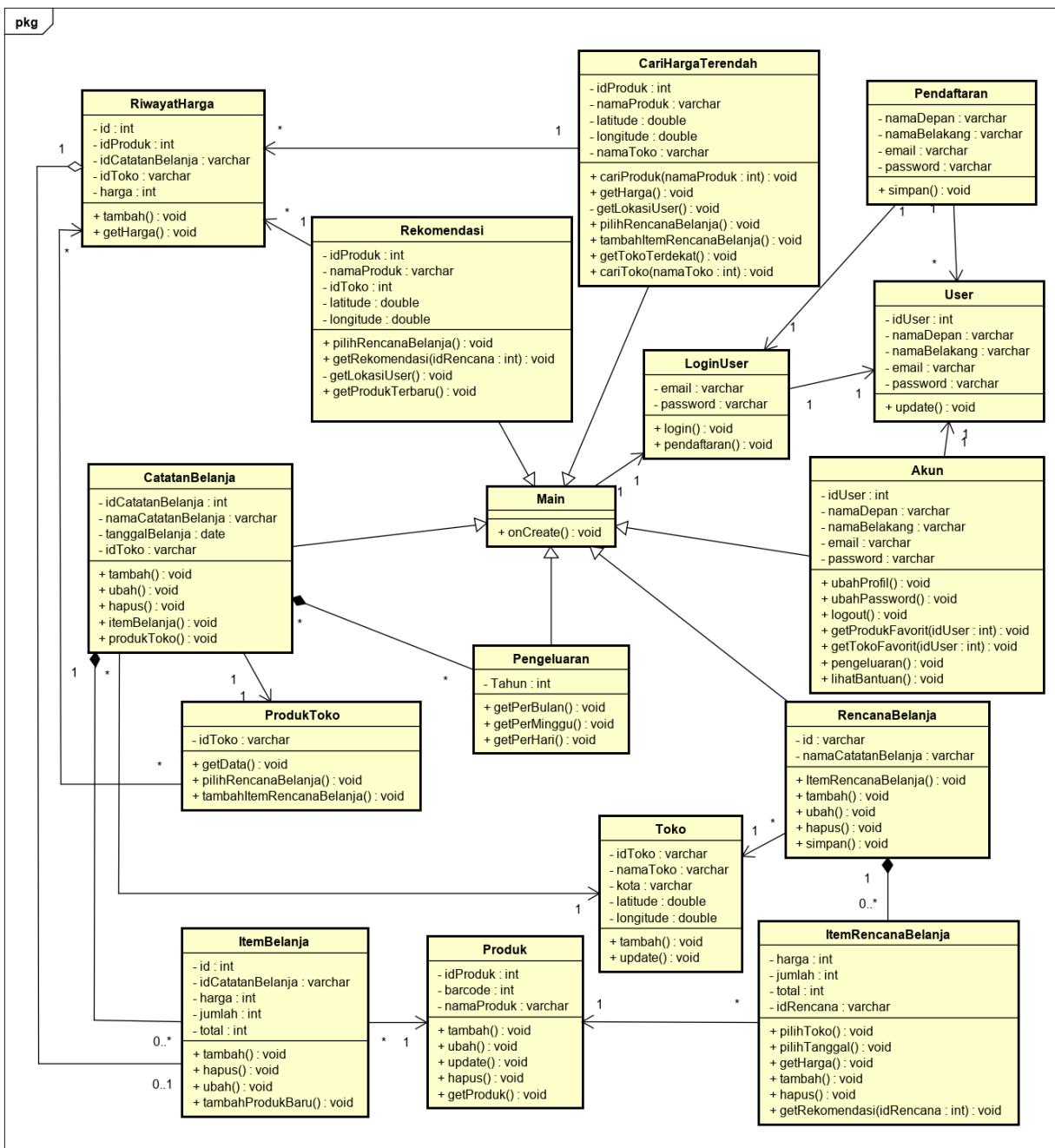


Gambar 3.18 *Activity Diagram Pengeluaran*

Pada Gambar 3.18, dapat dilihat bahwa *user* dapat melihat pengeluaran perhari, perminggu, dan perbulan pada halaman pengeluaran dengan memilih menu pengeluaran. Selanjutnya aplikasi akan menampilkan halaman pengeluaran perhari, perminggu, atau perbulan sesuai pilihan menu.

3.2.4.6 Perancangan Class Diagram

Class diagram merupakan salah satu diagram utama dari UML untuk menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem serta memiliki atribut dan metode atau operasi. Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pada tahap pengkodean dapat dibuat program perangkat lunak sesuai dengan perancangan diagram kelas.



Gambar 3.19 Class Diagram Aplikasi

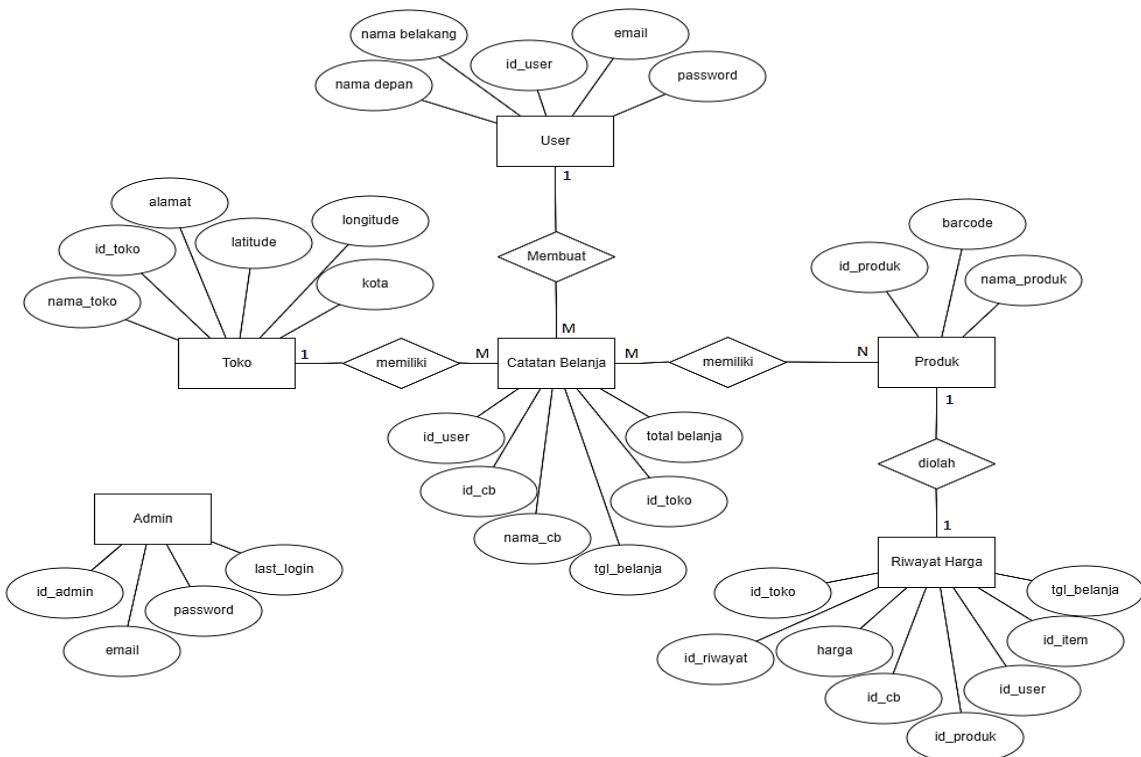
3.2.4.7 Perancangan Sequence Diagram

Sequence diagram adalah diagram yang menjelaskan interaksi objek berdasarkan urutan waktu. *Sequence diagram* menggambarkan urutan atau tahapan yang harus dilakukan pengguna. Diagram ini menunjukkan sejumlah contoh objek dan pesan yang melakukan satu tugas atau aksi tertentu. Komponen utama *sequence diagram* terdiri atas objek yang dituliskan dengan kotak persegi. Pesan diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress bar.

vertikal. Gambar *Sequence Diagram* dapat dilihat pada lampiran A.

3.2.4.8 Perancangan Basis Data

Berdasarkan perancangan *class diagram* maka didapatkan model data yang digunakan untuk mendefinisikan dan menganalisis kebutuhan data yang diperlukan oleh sistem. Pada penelitian ini digunakan model data berbasis obyek yaitu *Entity Relationship Model* yang digambarkan dengan *Entity Relation Diagram*. *Entity Relation Diagram* merupakan suatu teknik grafis yang menggambarkan skema basis data. Diagram ini terdiri dari sekumpulan objek dan relasi antar objek tersebut, serta dapat digunakan untuk menggambarkan relasi antara dua entitas atau lebih. Secara umum gambaran basis data yang akan dibuat sebagai berikut :



Gambar 3.20 Entity Relation Diagram Sistem

Berdasarkan *Entity Relation Diagram* maka didapatkan 7 tabel pada sistem ini yang dapat dilihat pada rincian berikut.

a. Spesifikasi Tabel User

Nama *database* : catatanbelanja

Nama tabel : user

Keterangan : tabel yang digunakan untuk menyimpan data *user* yang merupakan pengguna aplikasi.

Tabel 3.4 Spesifikasi Tabel User

Nama Field	Tipe	Boleh Kosong	Keterangan
id_user	<i>int</i> (10)	Tidak	Kunci Primer, Auto Increment.
nama_depan	<i>varchar</i> (30)	Tidak	Nama depan pengguna aplikasi
nama_belakang	<i>varchar</i> (50)	Tidak	Nama belakang pengguna aplikasi
email	<i>varchar</i> (100)	Tidak	Unique, Email pengguna aplikasi yang digunakan untuk login.
password	<i>varchar</i> (255)	Tidak	Password pengguna aplikasi, disimpan dalam bentuk hash md5.

b. Spesifikasi Tabel Admin

Nama *database* : catatanbelanja

Nama tabel : admin

Keterangan : tabel yang digunakan untuk menyimpan data admin web.

Tabel 3.5 Spesifikasi Tabel Admin

Nama Field	Tipe	Boleh Kosong	Keterangan
id_admin	<i>int</i> (10)	Tidak	Kunci Primer, Auto Increment.
email	<i>varchar</i> (100)	Tidak	Unique, Email admin yang digunakan untuk login ke web admin.
password	<i>varchar</i> (255)	Tidak	Password admin, disimpan dalam bentuk hash md5.
last_login	<i>timestamp</i>	Boleh	Tanggal terakhir kali admin login, dalam format ‘YYYY-MM-DD HH:MM:SS’

c. Spesifikasi Tabel Catatan Belanja

Nama *database* : catatanbelanja

Nama tabel : catatan_belanja

Keterangan : tabel yang digunakan untuk menyimpan catatan belanja milik *user*.

Tabel 3.6 Spesifikasi Tabel Catatan Belanja

Nama Field	Tipe	Boleh Kosong	Keterangan
id_catatan_belanja	<i>varchar</i> (100)	Tidak	Kunci Primer, merupakan kode unik yang dihasilkan aplikasi pada penyimpanan lokal.
id_user	<i>int</i> (10)	Tidak	Data sesuai dengan id_user pada tabel user.
tanggal_belanja	<i>date</i>	Tidak	Tanggal dengan format ‘YYYY-MM-DD’.
nama_catatan_belanja	<i>varchar</i> (30)	Tidak	Merupakan judul catatan belanja.
total_belanja	<i>int</i> (10)	Boleh	Berisi angka dengan nilai <i>default</i> 0.
id_toko	<i>varchar</i> (255)	Boleh	Data sesuai dengan id_toko pada tabel toko.

d. Spesifikasi Tabel Item Catatan Belanja

Nama *database* : catatanbelanja

Nama tabel : item_catatan_belanja

Keterangan : tabel yang digunakan untuk menyimpan data item catatan belanja.

Tabel 3.7 Spesifikasi Tabel Item Catatan Belanja

Nama Field	Tipe	Boleh Kosong	Keterangan
id_item	<i>int</i> (10)	Tidak	Kunci Primer, Auto Increment.
id_catatan_belanja	<i>varchar</i> (100)	Tidak	Data sesuai dengan id_catatan_belanja pada tabel catatan belanja.
id_produk	<i>int</i> (11)	Tidak	Data sesuai dengan id_produk pada tabel produk.
harga	<i>int</i> (11)	Tidak	Harga satuan produk.
jumlah	<i>int</i> (11)	Tidak	Jumlah pembelian produk.
total	<i>int</i> (11)	Tidak	Total harga dikali jumlah pembelian produk.

e. Spesifikasi Tabel Riwayat Harga

Nama *database* : catatanbelanja

Nama tabel : riwayat_harga

Keterangan : tabel yang digunakan untuk menyimpan data harga yang digunakan sebagai rekomendasi harga terendah.

Tabel 3.8 Spesifikasi Tabel Riwayat Harga

Nama Field	Tipe	Boleh Kosong	Keterangan
id	<i>int</i> (11)	Tidak	Kunci Primer, Auto Increment.
id_produk	<i>int</i> (11)	Tidak	Data sesuai dengan id_produk pada tabel produk.
id_user	<i>int</i> (11)	Tidak	Data sesuai dengan id_user pada tabel user.
id_catatan_belanja	<i>varchar</i> (100)	Tidak	Data sesuai dengan id_catatan_belanja pada tabel catatan belanja.
id_item	<i>int</i> (11)	Tidak	Data sesuai dengan id_item pada tabel item catatan belanja.
id_toko	<i>varchar</i> (255)	Tidak	Data sesuai dengan id_toko pada tabel toko.
tgl_belanja	<i>date</i>	Tidak	Tanggal belanja sesuai pada catatan belanja.
harga	<i>int</i> (11)	Tidak	Harga satuan produk.

f. Spesifikasi Tabel Produk

Nama *database* : catatanbelanja

Nama tabel : produk

Keterangan : tabel yang digunakan untuk menyimpan data produk.

Tabel 3.9 Spesifikasi Tabel Produk

Nama Field	Tipe	Boleh Kosong	Keterangan
id_produk	<i>int</i> (10)	Tidak	Kunci Primer, Auto Increment.
barcode	<i>varchar</i> (13)	Tidak	Barcode produk.
nama_produk	<i>varchar</i> (100)	Tidak	Nama produk.

g. Spesifikasi Tabel Toko

Nama *database* : catatanbelanja

Nama tabel	: toko
Keterangan	: tabel yang digunakan untuk menyimpan data toko.

Tabel 3.10 Spesifikasi Tabel Toko

Nama Field	Tipe	Boleh Kosong	Keterangan
id_toko	<i>varchar</i> (255)	Tidak	Kunci Primer, Berisi place id dari google places.
nama_toko	<i>varchar</i> (100)	Tidak	Nama tempat belanja.
alamat	<i>varchar</i> (200)	Tidak	Alamat tempat belanja.
kota	<i>varchar</i> (50)	Tidak	Kota tempat belanja berada.
lat	<i>double</i>	Tidak	Latitude tempat belanja.
lng	<i>double</i>	Tidak	Longitude tempat belanja.

3.2.5 Perancangan Antarmuka Sistem

Antarmuka merupakan tampilan dari suatu program aplikasi yang berperan sebagai sarana interaksi antara program dengan *user*. Sistem yang dibangun diharapkan dapat menyediakan antarmuka yang mudah dimengerti dan digunakan oleh *user*. Berikut ini adalah perancangan antarmuka perancangan pada sistem yang akan dibuat.

3.2.5.1 Perancangan Struktur Antarmuka Sistem

Adapun struktur antarmuka aplikasi catatan belanja dijabarkan sebagai berikut.

a) Menu Utama

Menu utama terdiri dari 4 buah submenu yakni menu catatan, rekomendasi, menu cari harga produk, dan menu akun.

b) Menu Catatan

Menu catatan terdiri dari 3 buah submenu yakni lihat catatan belanja yang berisikan data catatan belanja *user*, lihat rencana belanja yang berisikan data rencana belanja *user*, dan lihat pengeluaran yang menampilkan total pengeluaran perbulan, perminggu, dan perhari berdasarkan catatan belanja *user*.

c) Menu Rekomendasi

Menu rekomendasi merupakan menu yang mengizinkan *user* untuk

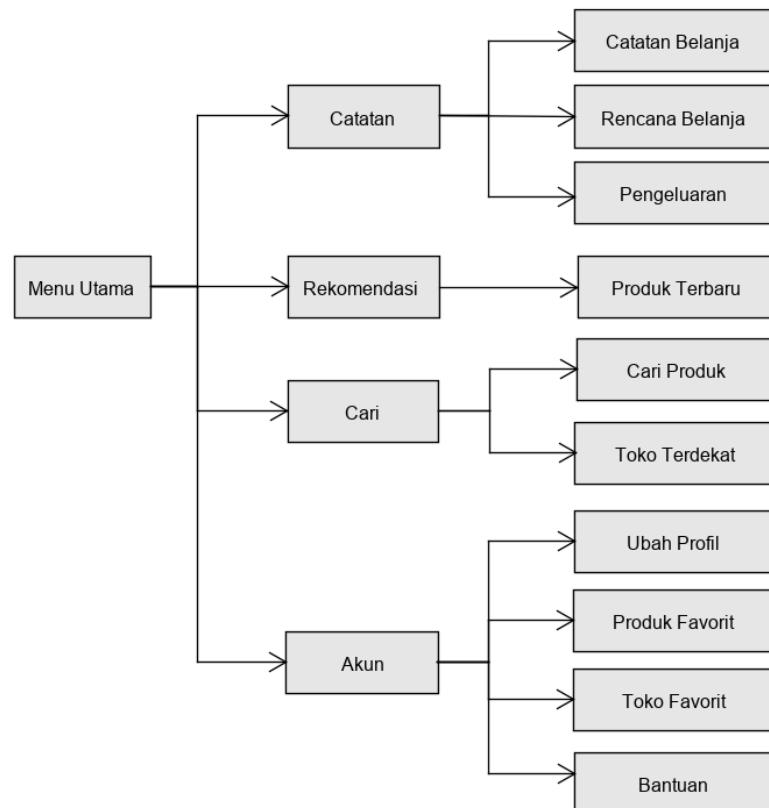
mendapatkan rekomendasi dengan cara memilih rencana belanja yang ingin dilihat rekomendasinya. Pada menu rekomendasi, *user* dapat melihat kota tempat user berada dan terdapat beberapa daftar produk terbaru.

d) Menu Cari Harga

Menu cari harga merupakan menu yang mengizinkan *user* untuk mencari produk dan kemudian melihat harga terendah dari berbagai toko disekitar *user* berdasarkan produk yang telah dipilih.

e) Menu Akun

Menu akun terdiri dari 2 buah submenu yakni ubah profil dan bantuan, serta terdapat daftar produk favorit dan toko favorit *user*.



Gambar 3.21 Perancangan Struktur Antarmuka Sistem

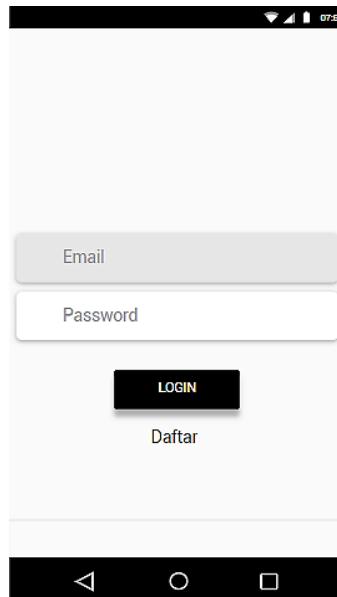
3.2.5.2 Perancangan *Layout*

Layout merupakan tata letak dari suatu elemen desain yang ditempatkan dalam sebuah bidang menggunakan sebuah media. Perancangan *layout* dilakukan untuk mengatur desain antarmuka aplikasi agar dapat mudah digunakan oleh *user*. Perancangan dilakukan dengan menggunakan Moqups

yang merupakan aplikasi web untuk pembuatan *wireframe*, *mockup*, dan *prototype*.

3.2.5.2.1 Perancangan Aktivitas Login

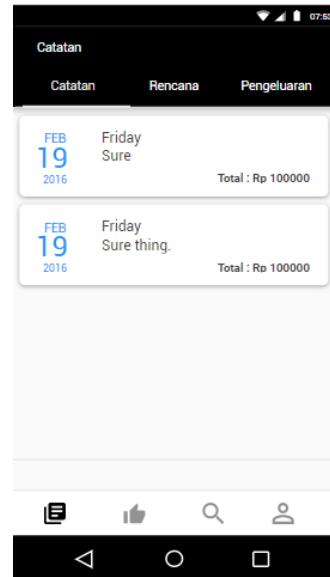
Aktivitas login adalah aktivitas yang ditampilkan pertama kali sebelum *user* dapat mengakses menu utama. Pada aktivitas login, *user* diminta memasukkan *email* dan *password*. Rancangan aktivitas login dapat dilihat pada Gambar 3.22.



Gambar 3.22 Rancangan Aktivitas Login

3.2.5.2.2 Perancangan Aktivitas Menu Catatan

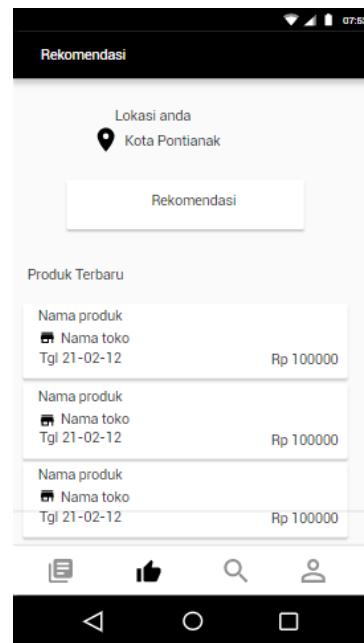
Aplikasi dirancang dengan 4 submenu yang dapat diakses pada *Bottom Navigation View* yakni menu catatan, rekomendasi, cari harga dan menu akun. Pada menu utama aplikasi secara *default* akan menampilkan menu catatan. Aktivitas catatan memuat 3 submenu yang ditampilkan dengan *Tab Layout*. *Tab* catatan belanja memuat daftar catatan belanja *user*, *tab* rencana belanja memuat daftar rencana belanja *user* yang ditampilkan berdasarkan data pada penyimpanan lokal, dan *tab* pengeluaran memuat total pengeluaran perbulan, perminggu, dan perhari. Rancangan aktivitas menu catatan dapat dilihat pada gambar 3.23.



Gambar 3.23 Rancangan Aktivitas Menu Catatan

3.2.5.2.3 Perancangan Aktivitas Rekomendasi

Aktivitas Rekomendasi adalah aktivitas yang menampilkan tombol untuk mengakses rekomendasi. Pada aktivitas rekomendasi, *user* juga dapat melihat daftar harga produk terbaru yang baru ditambahkan pada sistem. Rancangan aktivitas rekomendasi dapat dilihat pada Gambar 3.24.

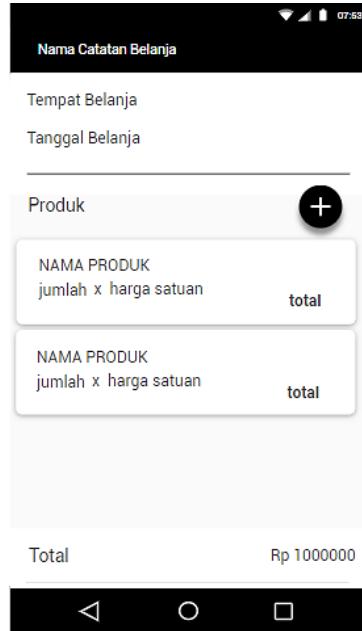


Gambar 3.24 Rancangan Aktivitas Rekomendasi

3.2.5.2.4 Perancangan Aktivitas Detail Catatan

Aktivitas detail catatan memuat informasi catatan meliputi tempat belanja, tanggal belanja, daftar item belanja, dan total belanja. Selain itu pada aktivitas detail

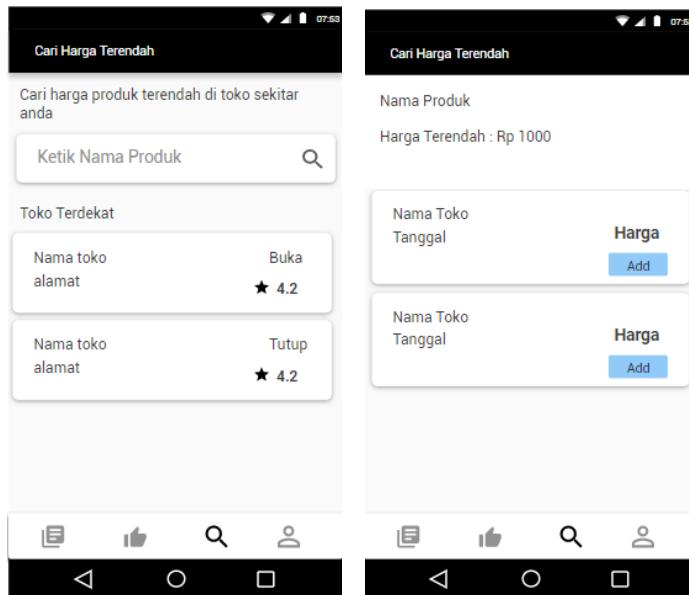
catatan terdapat menu yang digunakan untuk menambahkan item belanja. Rancangan aktivitas detail catatan dapat dilihat pada gambar 3.25.



Gambar 3.25 Rancangan Aktivitas Catatan

3.2.5.2.5 Perancangan Aktivitas Cari Harga

Aktivitas cari harga memuat fitur pencarian produk dan kemudian *user* dapat memilih produk yang diinginkan untuk ditampilkan daftar harga terendah produk tersebut yang akan diurutkan berdasarkan harga terendah. Rancangan aktivitas cari harga dapat dilihat pada gambar 3.26.



Gambar 3.26 Rancangan Aktivitas Cari Harga

3.2.6 Pembuatan Sistem

Aplikasi yang dibuat dapat berjalan pada sistem operasi android dengan minimum SDK 17 atau Android 4.2 (*Jelly Bean*) dan maksimum SDK 28 atau Android 9.0 (*Pie*). Aplikasi dibuat mengikuti rancangan dan dikembangkan dengan menggunakan Android Studio sebagai *Integrated Development Environment* (IDE) dengan bahasa pemrograman java dan *markup language* xml. Dalam pengembangan ada beberapa *library* yang digunakan mendukung pengembangan aplikasi. *Material Design* sebagai pendukung pembuatan antarmuka aplikasi untuk menghasilkan tampilan yang *simple/flat*. *Material Design* dapat diimplementasikan pada Android mulai dari versi 2.3.3 (*Gingerbread*) dan versi diatasnya dengan menambahkan *library appcompat v7* pada daftar dependensi aplikasi. Retrofit 2.0 sebagai pendukung komunikasi HTTP dengan webservice menggunakan *library OkHttp*. JSON merupakan format yang digunakan untuk pertukaran data pada webservice sehingga diperlukan *library Gson* untuk mengubah JSON menjadi *Object* sehingga pada pembuatan aplikasi tidak diperlukan lagi mengambil dan memindahkan data ke dalam model secara manual. Selain itu untuk mempermudah pengkodean aplikasi digunakan *Butterknife* untuk menyederhanakan penulisan komponen *View* pada *layout* yang telah dibuat sehingga mengurangi redundansi kode.

Pada pembuatan *barcode scanner* digunakan library dari Avaneesh Maurya yang dibangun menggunakan Google Mobile Vision API yang dapat diimplementasikan pada sistem operasi Android dengan minimum SDK 16 atau Android 4.1 (*Jelly Bean*). Webservice dikembangkan dengan bahasa pemrograman PHP berbantu Codeigniter Framework. *Library Codeigniter Rest Server* digunakan untuk mempermudah dalam pembangunan *RESTful webservice* pada Codeigniter Framework.

3.2.6.1 Perancangan Rekomendasi Harga Terendah

Pada pembuatan fitur harga terendah data id produk, latitude, dan longitude user dibutuhkan untuk dapat menghasilkan harga terendah dari berbagai tempat belanja yang sesuai dengan lokasi user. Pada saat terjadi permintaan harga terendah oleh aplikasi, akan dikirimkan data yang dibutuhkan untuk mendapatkan harga terendah secara bersamaan. Kemudian dilakukan penyortiran pada tabel riwayat

harga untuk mendapatkan harga terbaru dari semua tempat belanja terdekat. Setelah didapatkan harga terbaru dari setiap tempat belanja terdekat, data tersebut kemudian diurutkan berdasarkan harga dan toko dengan jarak terdekat sehingga dihasilkan harga terendah. Jarak terdekat dihasilkan dengan melakukan perhitungan menggunakan Formula Haversine.

Dengan mengasumsikan bahwa bumi berbentuk bulat sempurna dengan jari-jari 6371 km dan lokasi dari 2 titik di koordinat bola (lintang dan bujur) masing-masing adalah lng_1 , lat_1 , dan lng_2 , lat_2 , maka rumus Haversine dapat ditulis dengan persamaan sebagai berikut:

$$\begin{aligned}\Delta\text{lat} &= \text{lat}_2 - \text{lat}_1 \\ \Delta\text{lng} &= \text{lng}_2 - \text{lng}_1 \\ a &= \sin^2(\Delta\text{lat}/2) + \cos(\text{lat}_1) * \cos(\text{lat}_2) * \sin^2(\Delta\text{lng}/2) \\ c &= 2 \cdot \text{atan2}(\sqrt{a}, \sqrt{1-a}) \\ d &= R \cdot c\end{aligned}\tag{3.1}$$

Keterangan :

R = jari-jari bumi sebesar 6371(km)

Δlat = besaran perubahan *latitude*

Δlng = besaran perubahan *longitude*

c = kalkulasi perpotongan sumbu

d = jarak (km)

1 radian = 57.2957795 derajat

Berdasarkan formula haversine diatas dapat digunakan *query* SQL untuk menghasilkan jarak berdasarkan data yang ada pada basis data. Berikut ini merupakan *query* SQL yang dapat menghasilkan lokasi terdekat yang menghitung jarak antara *latitude* dan *longitude* toko yang ada pada basis data dan *latitude* dan *longitude* pengguna aplikasi. Untuk menghasilkan jarak dalam satuan mili R dapat diganti dengan 3959 atau 6371 untuk menghasilkan jarak dalam satuan kilometer (Google Developers, n.d.).

```
SELECT id, (acos(sin($user_lat) * sin(lat) + cos($user_lat) * cos(lat)
* cos(lng - ($user_lng))) * $R) AS distance FROM markers
```

(3.2)

3.2.7 Pengujian Sistem

Pengujian pada aplikasi dilakukan dengan menggunakan metode Black Box melalui sistem milik Bitbar Testing yang akan menguji fungsionalitas, performa, penggunaan memori, dan kompatibilitas, kemudian dilakukan pengujian jarak terdekat, pengujian *sample testing* untuk menguji harga terendah, dan kuesioner untuk mengetahui tingkat keberhasilan pembuatan aplikasi.

3.2.7.1 Pengujian Harga Terendah

Pengujian harga terendah dilakukan untuk menguji keakuratan aplikasi dalam menghasilkan harga terendah produk. Pengujian ini menggunakan skenario dengan data produk yang dibeli *user* pada tanggal yang berbeda dan tempat yang berbeda. Dari hasil inputan tersebut, akan dicek apakah sistem menghasilkan nilai harga terendah yang sesuai dengan yang diharapkan. Contoh skenario pengujian harga terendah adalah sebagai berikut:

Produk yang akan dicari harga terendahnya adalah chitato sapi panggang 40gr. Pada basis data, terdapat 6 data harga chitato sapi panggang 40gr dengan 3 toko berbeda yang dapat dilihat pada tabel 3.11.

Tabel 3.11 Contoh data harga yang tersedia pada tabel riwayat harga

No.	Nama Produk	Tanggal Belanja	Harga	Nama Toko
1	Chitato Sapi Panggang 40GR	10-03-2019	7000	Ligo Mitra
2	Chitato Sapi Panggang 40GR	12-03-2019	8000	Ligo Mitra
3	Chitato Sapi Panggang 40GR	08-03-2019	7000	Kaisar
4	Chitato Sapi Panggang 40GR	05-03-2019	10000	Kaisar
5	Chitato Sapi Panggang 40GR	10-03-2019	7500	Indomaret Gajahmada

No.	Nama Produk	Tanggal Belanja	Harga	Nama Toko
6	Chitato Sapi Panggang 40GR	12-03-2019	11000	Indomaret Gajahmada

Kriteria hasil :

1. Harga produk yang diambil pada satu toko merupakan harga produk yang terendah.
2. Jika terdapat harga produk pada toko yang sama dengan tanggal lebih baru namun bukan merupakan harga yang terendah maka harga produk yang lebih baru yang akan digunakan.
3. Setelah harga terendah produk dari masing-masing toko didapatkan, harga produk akan diurutkan mulai dari harga terendah ke harga tertinggi.

Hasil harga terendah yang diharapkan merupakan hasil yang memenuhi kriteria diatas dan merupakan data yang akan ditampilkan pada aplikasi, dapat dilihat pada tabel 3.12.

Tabel 3.12 Hasil harga terendah sesuai kriteria

No.	Nama Produk	Tanggal Belanja	Harga	Nama Toko
1	Chitato Sapi Panggang 40GR	08-03-2019	7000	Kaisar
2	Chitato Sapi Panggang 40GR	12-03-2019	8000	Ligo Mitra
3	Chitato Sapi Panggang 40GR	12-03-2019	11000	Indomaret Gajahmada

3.2.7.2 Pengujian Jarak Terdekat

Pengujian jarak terdekat dilakukan dengan membandingkan hasil perhitungan jarak yang dihasilkan oleh sistem dan hasil perhitungan jarak dari Google Maps. Perbandingan dilakukan dengan menggunakan lokasi *default* Gedung Informatika Untan, Pontianak dengan koordinat -0.0556337,109.3463961. Kemudian ditentukan beberapa daftar tempat untuk dilakukan pengujian. Hasil

perhitungan jarak yang dihasilkan aplikasi dan hasil perhitungan jarak dari google maps kemudian dihitung selisih perbedaan hasilnya dan dirangkum dalam sebuah tabel seperti pada tabel 3.13 berikut.

Tabel 3.13 Perbandingan Hasil Perhitungan Jarak Terdekat

No	Nama Toko	Jarak Haversine (Km)	Jarak Google Maps (Km)	Selisih
1				
2				
3				
4				

3.2.7.3 Pengujian Blackbox

Pengujian fungsional yang digunakan untuk menguji sistem yang baru adalah metode pengujian alpha. Metode yang digunakan dalam pengujian ini adalah pengujian *black box* yang berfokus pada fungsional, performa, dan kompatibilitas aplikasi yang dibangun. Pengujian *black box* digunakan untuk menguji validitas dari integrasi dan konsistensi sistem.

3.2.7.4 Pengujian Kuesioner

Pengujian kuesioner yaitu menguji tingkat penerimaan pengguna terhadap sistem melalui kuesioner. Kuesioner berisi beberapa pertanyaan yang dikelompokkan menjadi 3 aspek yang digunakan dalam pengujian aplikasi tersebut, yaitu aspek rekayasa perangkat lunak, aspek fungsionalitas, dan aspek komunikasi visual. Kuesioner dibagikan kepada 30 responden yang merupakan pengguna Android. Responden akan mencoba aplikasi untuk dapat mengetahui kapabilitas dan kelancaran jalannya aplikasi itu sendiri. Hasil kuesioner dirangkum berdasarkan tiga aspek yang diujikan, yaitu.

3.2.7.4.1 Aspek Rekayasa Perangkat Lunak

Kuesioner aspek rekayasa perangkat lunak dirangkum dalam sebuah tabel seperti pada tabel 3.14 berikut.

Tabel 3.14 Kuesioner Aspek Rekayasa Perangkat Lunak

No	Aspek Rekayasa Perangkat Lunak	Tanggapan					Total
		1	2	3	4	5	
1	Kemudahan menjalankan aplikasi						
2	Kompatibilitas aplikasi pada perangkat						
3	Kelancaran menjalankan aplikasi pada perangkat						
4	Kemudahan mengakses fitur – fitur pada aplikasi						
5	Kenyamanan dalam penggunaan aplikasi secara keseluruhan						
Jumlah							
Percentase (%)							

Keterangan : 1 = Sangat Buruk 2 = Buruk 3 = Cukup baik
 4 = Baik 5 = Sangat Baik

3.2.7.4.2 Aspek Fungsionalitas

Kuesioner aspek fungsionalitas dirangkum dalam sebuah tabel seperti pada tabel 3.15 berikut.

Tabel 3.15 Kuesioner Aspek Fungsionalitas

No	Aspek Fungsionalitas	Tanggapan					Total
		1	2	3	4	5	
1	Kinerja aplikasi saat menampilkan data						
2	Tingkat kesesuaian menampilkan hasil harga produk terendah						
3	Kinerja aplikasi saat melakukan manajemen data						
Jumlah							

No	Aspek Fungsionalitas	Tanggapan					Total
		1	2	3	4	5	
	Persentase (%)						

Keterangan : 1 = Sangat Buruk 2 = Buruk 3 = Cukup baik
 4 = Baik 5 = Sangat Baik

3.2.7.4.3 Aspek Komunikasi Visual

Kuesioner aspek komunikasi visual dirangkum dalam sebuah tabel seperti pada tabel 3.16 berikut.

Tabel 3.16 Kuesioner Aspek Komunikasi Visual

No	Aspek Komunikasi Visual	Tanggapan					Total
		1	2	3	4	5	
1	Tampilan (antarmuka) aplikasi						
2	Jenis dan ukuran huruf yang digunakan mudah dibaca						
3	Kombinasi warna pada tampilan aplikasi						
4	Respon (<i>feedback</i>) aplikasi terhadap input yang dimasukkan						
5	Kemudahan memahami informasi yang ditampilkan pada aplikasi						
Jumlah							
Persentase (%)							

Keterangan : 1 = Sangat Buruk 2 = Buruk 3 = Cukup baik
 4 = Baik 5 = Sangat Baik

Hasil kuesioner yang telah dirangkum berdasarkan aspek pengujian tersebut akan dihitung jumlah dan persentase (%) dari tanggapan seluruh responde. Dari persentase yang telah didapat, akan dibuat diagram lingkaran sebagai visualisasi setiap aspek hasil kuesioner.

Untuk mengetahui tingkat keberhasilan aplikasi dari kuesioner, digunakan

metode Likert's Summated Rating (LSR) untuk mengukur skor terkecil dan terbesar dari 13 pertanyaan kuesioner yang ditanggapi oleh responden. Skor tanggapan dari 13 pertanyaan untuk setiap responden dirangkum dalam sebuah tabel seperti pada Tabel 3.17 berikut.

Tabel 3.17 Total skor responden

Responden	Item													Total
	1	2	3	4	5	6	7	8	9	10	11	12	13	
25														
26														
27														
28														
29														
30														
Total Skor														

Data yang diperoleh dari hasil pengujian dengan kuesioner kemudian diukur dengan metode *Likert's Summated Rating* (LSR).

1. Jumlah skor untuk setiap responden:
 - Skor maksimal = 65 (5 x 13 item)
 - Skor minimal = 13 (1 x 13 item)
 - Skor median = 39 (3 x 13 item)
 - Skor kuartil I = 26 (2 x 13 item)
 - Skor kuartil III = 52 (4 x 13 item)
2. Jumlah skor untuk seluruh responden:
 - Maksimal = 1950 (30 x 65)
 - Minimal = 390 (30 x 13)
 - Median = 1170 (30 x 39)
 - Kuartil I = 780 (30 x 26)
 - Kuartil II = 1560 (30 x 52)
3. Interpretasi jumlah skor :
 - $1560 < \text{skor} < 1950$, artinya aplikasi dinilai sangat positif (program dinilai berhasil)
 - $1170 < \text{skor} < 1560$, artinya aplikasi dinilai positif (program dinilai cukup berhasil)
 - $780 < \text{skor} < 1170$, artinya aplikasi dinilai negatif (program dinilai kurang berhasil)

- $390 < \text{skor} < 780$, artinya aplikasi dinilai sangat negatif (program dinilai tidak berhasil)

3.2.8 Analisa Hasil Pengujian

Pada analisa hasil pengujian, dilakukan analisis terhadap sistem secara keseluruhan berdasarkan pengujian yang sudah dilakukan.

3.2.9 Penarikan Kesimpulan

Kesimpulan dirumuskan berdasarkan analisis hasil pengujian sistem yang dilakukan apakah sistem yang dirancang sesuai dengan tujuan yang ingin dicapai dalam penelitian.

BAB IV

IMPLEMENTASI DAN HASIL PENGUJIAN

4.1 Implementasi

Hasil implementasi dari hasil perancangan menghasilkan sebuah aplikasi android yaitu aplikasi catatan belanja dan sebuah web admin untuk memanajemen data produk yang dibuktikan dengan tangkapan layar (*screenshot*) dari aplikasi dan web yang dibuat.

4.1.1 Rekomendasi Harga Terendah Per-total

Rekomendasi harga terendah diolah dari beberapa data yaitu data koordinat geografis pengguna aplikasi berupa *latitude* dan *longitude*, dan data id produk. Output yang dihasilkan oleh sistem yaitu rekomendasi tempat belanja dengan kriteria berdasarkan ketersediaan produk, total harga terendah, dan jarak toko terdekat. Berikut ini merupakan langkah yang dilakukan untuk menghasilkan rekomendasi, yaitu :

1. *Request* lokasi *user*, pada langkah ini aplikasi akan *me-request* lokasi *user* menggunakan GPS.
2. Kumpulkan semua id produk rencana belanja, pada tahap ini aplikasi akan mengambil semua id produk rencana belanja yang ingin dilihat rekomendasinya oleh *user*.
3. Konversi koordinat *user*, pada langkah ini terdapat proses konversi *latitude* dan *longitude user* dalam satuan derajat menjadi radians.
4. Filter data harga produk, pada langkah ini akan diambil semua data harga terbaru dari produk yang dicari, jika terdapat data dengan tanggal dan id produk sama pada toko yang sama maka akan diambil data dengan harga terendah.
5. Kalkulasi Haversine, pada langkah ini data harga yang sudah di proses kemudian akan diambil data hanya pada toko yang berjarak tidak lebih dari 8 km dari lokasi *user*. Proses perhitungan ini dilakukan dengan menggunakan formula Haversine.
6. Hitung total harga dan jumlah produk, pada langkah ini data harga yang sudah diproses kemudian akan dihitung total harga dan jumlah produk yang tersedia dengan cara dikelompokkan berdasarkan toko.

7. Tampilkan daftar toko, pada langkah ini aplikasi menampilkan hasil memrosesan data yang telah diurutkan berdasarkan ketersediaan produk terbanyak, total harga terendah, dan jarak terdekat.

Berikut contoh analisis cara kerja metode Haversine Formula dalam perhitungan jarak antara dua titik:

1. Titik koordinat pertama

(Kaisar Supermarket And Department Store)

$$\begin{aligned} \text{Latitude 1} &= -0.02818730000000002 / 57.2957795 \\ &= -0.00049196119 \text{ Radian} \end{aligned}$$

$$\begin{aligned} \text{Longitude 1} &= 109.3390434999999 / 57.2957795 \\ &= 1.9083263 \text{ Radian} \end{aligned}$$

2. Titik koordinat kedua

(User: Gedung Informatika Untan)

$$\begin{aligned} \text{Latitude 2} &= -0.0556337 / 57.2957795 \\ &= -0.00097099124 \text{ Radian} \end{aligned}$$

$$\begin{aligned} \text{Longitude 2} &= 109.3463961 / 57.2957795 \\ &= 1.90845464 \text{ Radian} \end{aligned}$$

$$\begin{aligned} 3. \Delta\text{lat} &= -0.00097099124 - (-0.00049196119) \\ &= 0.00047903005 \end{aligned}$$

$$\begin{aligned} 4. \Delta\text{lng} &= 1.90845464 - 1.9083263 \\ &= 0.00012834 \end{aligned}$$

$$\begin{aligned} 5. a &= \sin^2(\Delta\text{lat}/2) + \cos(\text{lat1}) * \cos(\text{lat2}) * \sin^2(\Delta\text{lng}/2)/2 \\ &= \sin^2(0.00047903005/2) + \cos(-0.00049196119) * \\ &\quad \cos(-0.00097099124) * \sin^2(0.00012834/2) \\ &= 0.0000000614852326 \end{aligned}$$

$$\begin{aligned} 6. c &= 2 * a \sin \sqrt{0.0000000614852326} \\ &= 0.000495924314 \end{aligned}$$

$$\begin{aligned} 7. d &= R * c \\ &= 6371 * 0.000495924314 \\ &= 3.1595338 \text{ km} \end{aligned}$$

Hasil dari contoh perhitungan yaitu 3.16 km yang dihitung dari koordinat Kaisar Supermarket And Department Store ke Gedung Informatika Untan. Hasil

tersebut yang kemudian diurutkan dan dicari nilai yang terkecil dan tidak melebihi dari 10km.

Berikut adalah rumus haversine dalam *query SQL*:

$$(acos(sin(\$user_lat) * sin(lat) + cos(\$user_lat) * cos(lat) * cos(lng - ($user_lng))) * \$R) \quad (4.1)$$

Keterangan :

- \$R = jari-jari bumi sebesar 6371(km).
- \$user_lat = *latitude user* dalam satuan radian.
- \$user_lng = *longitude user* dalam satuan radian.
- lat = kolom dalam tabel database yang menyimpan data *latitude* sebuah tempat dalam radian.
- lng = kolom dalam tabel database yang menyimpan data *longitude* sebuah tempat dalam radian

Berdasarkan penjabaran langkah-langkah diatas kemudian didapatkan rumus yang direpresentasikan dalam *query SQL* sebagai berikut.

Kode Program 4.1 *Query SQL* Rekomendasi Harga Terendah Per-total

```

1  function getRekomendasi($idproduk, $userlat, $userlng, $orderby, $distance, $minjumlah)
2  {
3      $produk = explode(",", $idproduk);
4      if (empty($minjumlah)) {
5          $jumlah = count($produk);
6          if ($jumlah < 10 | $jumlah >=2) {
7              $minProduk = $jumlah-1;
8          } else if ($jumlah >10) {
9              $minProduk = $jumlah-2;
10         } else {
11             $minProduk = $jumlah;
12         }
13     } else {
14         $minProduk = $minjumlah;
15     }
16
17     $radians_to_degs = 57.2957795;
18     $current_lat = $userlat / $radians_to_degs;
19     $current_lon = $userlng / $radians_to_degs;
20     $earths_radius = 6371;
21     $sql =
22         "SELECT SUM(s.`harga`) as total, COUNT(s.`id_produk`) as jumlah_produk, s.`id_toko`,
23         s.`nama_toko`, s.`alamat`, s.`distance` FROM (
24             SELECT
25                 `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_produk`,
26                 `riwayat_harga`.`id_user`, `riwayat_harga`.`id_catatan_belanja`,
27                 `riwayat_harga`.`id_toko`, `riwayat_harga`.`tanggal_belanja`,
28                 `riwayat_harga`.`harga`, `toko`.`nama_toko`, `toko`.`kota`, `toko`.`alamat`, (acos(
29                     sin(`$this->db->escape_str($current_lat).") * sin(`toko`.`lat` /
30                     ".$this->db->escape_str($radians_to_degs).") +
31                     cos(`$this->db->escape_str($current_lat).") * cos(`toko`.`lat` /
32                     ".$this->db->escape_str($radians_to_degs).") * cos(`toko`.`lng` /
33                     ".$this->db->escape_str($radians_to_degs).") -
34                     (".$this->db->escape_str($current_lon).")) *
35                     ".$this->db->escape_str($earths_radius).") as distance FROM `riwayat_harga`
36             INNER JOIN (

```

```

26      SELECT `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_toko`,
27      `riwayat_harga`.`id_produk`, `riwayat_harga`.`tanggal_belanja`, MIN(`harga`) AS
28      MinHarga FROM `riwayat_harga`
29      INNER JOIN ( SELECT `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_produk`,
30      `riwayat_harga`.`id_toko`, MAX(`tanggal_belanja`) AS MaxTgl FROM
31      `riwayat_harga` WHERE `riwayat_harga`.`id_produk` IN
32      (".$this->db->escape_str($idproduk).") GROUP BY `riwayat_harga`.`id_produk`,
33      `id_toko`)
34      groupa ON `riwayat_harga`.`tanggal_belanja` = groupa.MaxTgl AND
35      `riwayat_harga`.`id_toko` = groupa.`id_toko` WHERE `riwayat_harga`.`id_produk`  

36      IN (".$this->db->escape_str($idproduk).") GROUP BY
37      `riwayat_harga`.`id_toko`, `riwayat_harga`.`id_produk`)
38      groupb ON `riwayat_harga`.`id_toko` = groupb.`id_toko` AND
39      `riwayat_harga`.`tanggal_belanja` = groupb.`tanggal_belanja` AND
40      `riwayat_harga`.`id_produk` = groupb.`id_produk` AND `riwayat_harga`.`harga` =
41      groupb.MinHarga AND `riwayat_harga`.`id_riwayat` = groupb.`id_riwayat`
42      INNER JOIN `toko` ON `riwayat_harga`.`id_toko` = `toko`.`id_toko` WHERE acos(
43      sin(".$this->db->escape_str($current_lat).") * sin(`toko`.`lat` /
44      ". $this->db->escape_str($radians_to_degs).") +
45      cos(".$this->db->escape_str($current_lat).") * cos(`toko`.`lat` /
46      ". $this->db->escape_str($radians_to_degs).") * cos(`toko`.`lng` /
47      ". $this->db->escape_str($radians_to_degs).") -
48      (".$this->db->escape_str($current_lon).")) * ".$this->db->escape_str($earths_radius)."
49      <= ".$this->db->escape_str($distance)." ORDER BY groupb.MinHarga ASC )
50      S GROUP BY s.`id_toko` HAVING COUNT(s.`id_produk`) >=
51      ".$this->db->escape_str($minProduk)." ORDER BY ".$this->db->escape_str($orderby);

52      $query = $this->db->query($sql)->result();
53      $hasilrekомendasi = array();

```

Kode program 4.1 merupakan proses yang terjadi setelah *user* mengirimkan *request* rekomendasi melalui aplikasi. Langkah yang pertama kali dilakukan yaitu memeriksa apakah pengguna menentukan minimal jumlah ketersediaan produk, jika tidak maka jumlah minimal ketersediaan produk akan ditentukan dengan memecah string id produk untuk dihitung jumlah produk yang akan dicari seperti yang terlihat pada baris 5 pada kode program.

Kemudian dilakukan penentuan jumlah minimal ketersediaan produk seperti yang terlihat pada baris 6 sampai baris 12. Jika jumlah produk yang diinputkan lebih dari sama dengan 2 dan kurang dari 10 maka minimal jumlah produk yang tersedia yaitu jumlah produk dikurangi 1. Jika jumlah produk lebih dari 10 maka minimal jumlah produk yang tersedia yaitu jumlah produk dikurangi 2, atau dengan kata lain rekomendasi yang ditampilkan hanya toko yang jumlah produk tidak tersedianya hanya satu atau dua barang saja dari seluruh jumlah produk. Jenis produk yang tidak tersedia pada setiap toko tidak dipertimbangkan sehingga akan ada produk tidak tersedia yang berbeda-beda pada setiap toko dan hasil rekomendasi yang ditampilkan tetap dilakukan dengan mengurutkan hasil dengan total harga terendah.

Pada baris 17 sampai baris 19, dilakukan konversi koordinat geografis yang

user yang sebelumnya dalam derajat menjadi radians yang merupakan persiapan data untuk perhitungan jarak dengan menggunakan formula haversine. Setelah semua data siap maka dilakukan permintaan ke *database* untuk mencari data harga produk terbaru dan terendah pada toko terdekat dengan jarak tidak lebih dari 8 km pada kode program baris 21 sampai dengan baris 31 untuk kemudian ditampilkan pada *web service* dalam format json dan ditampilkan ke aplikasi.

4.1.2 Rekomendasi Harga Terendah Per-produk

Rekomendasi harga terendah per-produk diimplementasikan dengan menggunakan kueri SQL pada database mySQL. Harga terendah yang diambil merupakan data terbaru yang ada pada basis data.

Kode Program 4.2 Kueri SQL Daftar Harga Terendah

```

1  public function getRiwayatHarga($userlat, $userlng, $idproduk)
2  {
3
4      $radians_to_degs = 57.2957795;
5      $distance = 10;
6      $current_lat = $userlat / $radians_to_degs;
7      $current_lon = $userlng / $radians_to_degs;
8      $earths_radius = 6371;
9      $sql = "SELECT
10          `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_produk`,
11          `riwayat_harga`.`id_user`, `riwayat_harga`.`id_catastan_belanja`,
12          `riwayat_harga`.`id_toko`, `riwayat_harga`.`tanggal_belanja`,
13          `riwayat_harga`.`harga`, `toko`.`nama_toko`, `toko`.`kota`, `toko`.`alamat`, (acos
14          (sin(`$current_lat.`) * sin(`toko`.`lat` / ".$radians_to_degs.") +
15          cos(`$current_lat.`) * cos(`toko`.`lat` / ".$radians_to_degs.") *
16          cos(`toko`.`lng` / ".$radians_to_degs.") - (".$current_lon.")) *
17          ".$earths_radius.) as distance FROM `riwayat_harga`
18          INNER JOIN (
19              SELECT `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_toko`,
20                  `riwayat_harga`.`id_produk`, `riwayat_harga`.`tanggal_belanja`,
21                  MIN(`harga`) AS MinHarga FROM `riwayat_harga`
22                  INNER JOIN (
23                      SELECT `riwayat_harga`.`id_riwayat`, `riwayat_harga`.`id_toko`,
24                          MAX(`tanggal_belanja`) AS MaxTgl FROM `riwayat_harga` WHERE
25                      `riwayat_harga`.`id_produk` = ".$this->db->escape($idproduk)." GROUP
26                      BY `id_toko` ) groupa
27                      ON `riwayat_harga`.`tanggal_belanja` = groupa.MaxTgl WHERE
28                      `riwayat_harga`.`id_produk` = ".$this->db->escape($idproduk)." GROUP BY
29                      `riwayat_harga`.`id_toko` ) groupb
30                      ON `riwayat_harga`.`id_toko` = groupb.`id_toko` AND
31                      `riwayat_harga`.`tanggal_belanja` = groupb.`tanggal_belanja` AND
32                      `riwayat_harga`.`id_produk` = groupb.`id_produk` AND `riwayat_harga`.`harga` =
33                      groupb.MinHarga
34                      INNER JOIN `toko` ON `riwayat_harga`.`id_toko` = `toko`.`id_toko` WHERE acos(
35                      sin(`$current_lat.`) * sin(`toko`.`lat` / ".$radians_to_degs.") +
36                      cos(`$current_lat.`) * cos(`toko`.`lat` / ".$radians_to_degs.") *
37                      cos(`toko`.`lng` / ".$radians_to_degs.") - (".$current_lon.")) *
38                      ".$earths_radius." <= ".$distance." ORDER BY groupb.MinHarga ASC";
39
40      $query = $this->db->query($sql)->result_array();
41      $response['result'] = $query;
42      $response['error'] = false;
43      return $response;
44  }

```

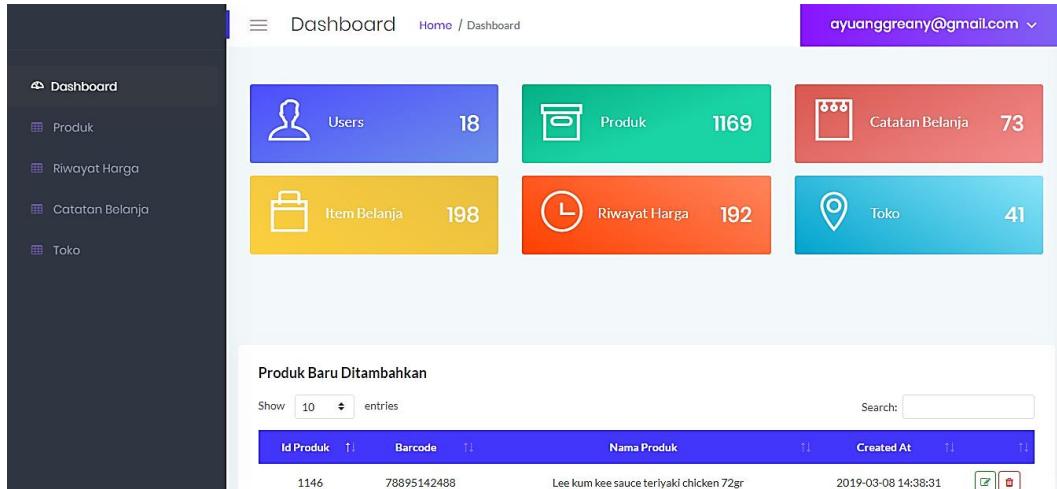
Data yang dibutuhkan untuk menghasilkan harga terendah yaitu id produk

yang akan dicari harganya, *latitude* dan *longitude user*. *Latitude* dan *longitude* digunakan untuk menyortir toko sehingga toko yang ditampilkan hanya toko yang berjarak tidak lebih dari 10km dari lokasi *user*. Data *latitude* dan *longitude user* dikirim dalam koordinat kemudian dikonversi ke radians pada kode baris 6 dan 7. Konversi ke radians diperlukan karena pada perhitungan jarak antara toko dan *user* dilakukan dalam radians.

Pada kueri, yang dilakukan pertama yaitu mengambil semua data pada tabel riwayat harga. Kemudian tabel riwayat harga dipilih data dengan harga terendah pada kode baris ke 12 dan digabungkan dengan tabel riwayat harga yang dipilih data harga produk yang dicari dengan tanggal belanja terbaru yang dikelompokkan berdasarkan toko pada kode program baris ke 14 menjadi sebuah tabel yang diberi nama groupb. Tabel groupb kemudian digabungkan dengan tabel riwayat harga yang berisi semua data riwayat harga menggunakan INNER JOIN dengan kondisi tanggal belanja, id produk, dan harga sama dan digabungkan dengan tabel toko juga untuk mendapatkan deskripsi toko seperti nama toko dan alamat. Data yang dihasilkan kemudian diurutkan berdasarkan harga mulai dari yang terendah sampai tertinggi, jika terdapat harga yang sama maka data hasil akan terurut berdasarkan toko dengan jarak terdekat.

4.1.3 Tampilan Antarmuka Halaman Dashboard Website

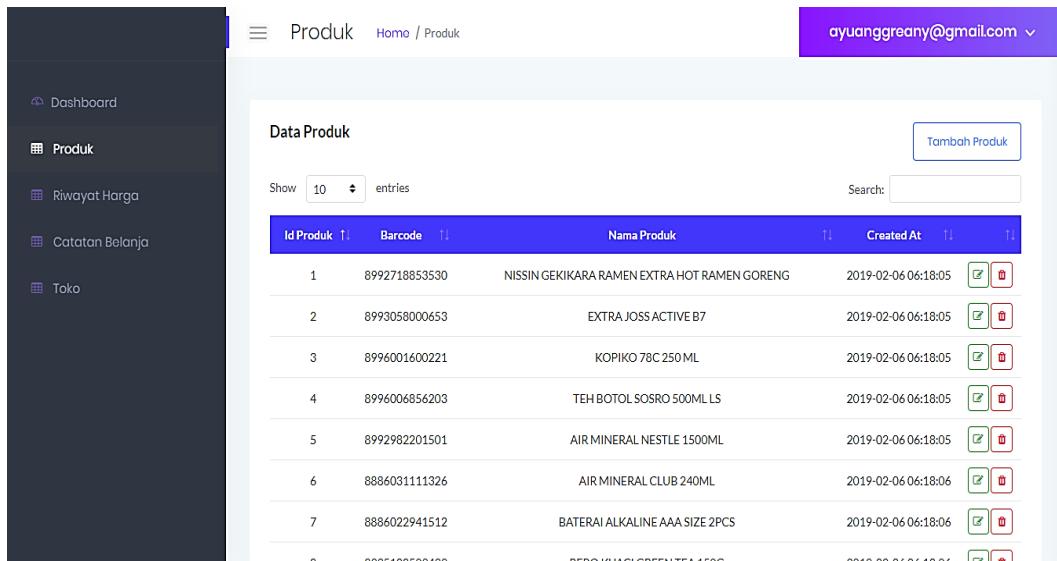
Pada tampilan halaman *dashboard* website admin dapat melihat informasi jumlah user aplikasi yang sudah terdaftar dan jumlah produk yang sudah ditambahkan. Selain itu admin dapat melihat data produk yang baru ditambahkan oleh user atau admin. Halaman ini hanya dapat diakses jika admin berhasil *login* dengan menggunakan *email* dan *password*. Antarmuka halaman *dashboard* dapat dilihat pada Gambar 4.1.



Gambar 4.1 Antarmuka Halaman *Dashboard*

4.1.4 Tampilan Antarmuka Halaman Produk

Pada halaman Produk, admin dapat melihat semua data produk, menambahkan produk baru, mengubah produk, dan menghapus produk yang sudah ditambahkan. Antarmuka halaman produk dapat dilihat pada Gambar 4.2.



Gambar 4.2 Antarmuka Halaman Produk

4.1.5 Tampilan Antarmuka Halaman Riwayat Harga

Pada halaman riwayat harga, admin dapat melihat semua data riwayat harga. Antarmuka halaman riwayat harga dapat dilihat pada Gambar 4.3.

ID	ID Produk	Nama Produk	Harga	Tanggal Belanja	ID Toko	Nama Toko	ID User
1	1016	NUTRIVE BENECOL 100ML ORANGE	7600	2018-08-30	ChJQXYqAiSYHS4RIPqXqHS6X8k	Indomaret Panglima Alm	1
2	1017	NUTRIVE BENECOL ECI 100ML	7600	2018-08-30	ChJQXYqAiSYHS4RIPqXqHS6X8k	Indomaret Panglima Alm	1
3	1018	MAMY POKO PANTS M 34 SHEET	87000	2018-08-30	ChJQXYqAiSYHS4RIPqXqHS6X8k	Indomaret Panglima Alm	1
4	1019	FRUIT TEA STRAWBERRY BTL 350ML	3500	2018-08-30	ChJQXYqAiSYHS4RIPqXqHS6X8k	Indomaret Panglima Alm	1
Total Belanja : 94.800							

Gambar 4.3 Antarmuka Halaman Riwayat Harga

4.1.6 Tampilan Antarmuka Catatan Belanja

Pada halaman catatan belanja, admin dapat melihat semua catatan belanja dan item catatan belanja yang sudah dimasukkan oleh pengguna aplikasi. Antarmuka halaman catatan belanja dan item catatan belanja dapat dilihat pada Gambar 4.4.

ID	ID User	Tanggal Belanja	Nama Catatan	Total Belanja	ID Toko	Nama Toko	Created At
17[B@43c92fa	17	2019-04-15	Wiwin	37250	ChJNVKQJKJRZHS4RpNw6RpFUU	Park Untan Fountain	2019-04-15 17:24:35
17[B@c07d3fa	17	2019-03-28	Harini	5000	ChJQ_VdxZVZHS4RcRHWizzL17A	D' Coffe	2019-03-28 16:36:18
17[B@c67d939	17	2019-03-25	Seminar Hasil	0	ChJRz3X_A9ZHS4Rx1aZJKTf-cg	Hypermart Mega Mall Pontianak	2019-03-28 17:03:58
18[B@759828f	18	2019-04-08	01	18000	ChJWbN9_qpZHS4Rt526O0WXJUc	Hypermart	2019-04-08 14:02:35
1[B@1096a4	1	2018-08-19	15	24100	ChJ3R1PGTJYHS4RvJJWAbcPCIQ	Sim Jaya Abadi	2019-03-05 14:22:01
Total Belanja : 44.800							

Gambar 4.4 Antarmuka Halaman Catatan Belanja

4.1.7 Tampilan Antarmuka Toko

Pada halaman toko, admin dapat melihat semua toko yang sudah dimasukkan oleh pengguna aplikasi melalui catatan belanja. Antarmuka halaman toko dapat dilihat pada Gambar 4.5.

Id	Nama Toko	Alamat	Latitude	Longitude	Created At
ChIJ1xAhtFIYHS4RMQm3VVp6NHA	Carrefour Pontianak	Matahari Mall, Jl. Kartini, Tengah, Pontianak Kota, Kota Pontianak, Kalimantan Barat 78243, Indonesia	-0.0248257	109.3367754	2019-03-06 14:52:09
ChIJ1yKadEtZHS4Rxp-wT5IQkO4	Indomaret Merdeka	Jl. Merdeka Barat, No.633B, Mariana, Pontianak Kota, Kota Pontianak, Kalimantan Barat 78244, Indonesia	-0.0218367	109.3294538	2019-03-06 14:52:09
		Jl. Tj. Raya II, Saigon, Pontianak			

Gambar 4.5 Antarmuka Halaman Toko

4.1.8 Tampilan Antarmuka Menu Utama

Pada menu utama yang terletak pada *bottom navigation view* terdapat 4 menu utama yaitu catatan, rekomendasi, cari, dan akun. Menu catatan merupakan menu yang pertama ditampilkan ketika aplikasi dibuka, pada menu catatan *user* dapat melihat catatan dan rencana belanja, kemudian *user* dapat membuat catatan belanja baru pada tombol tambah. Pada menu rekomendasi, *user* dapat melihat rekomendasi berdasarkan rencana belanja dan melihat produk terbaru yang ditambahkan oleh *user* lain. Pada menu cari harga, *user* dapat melihat harga produk terendah dari berbagai toko dan melihat daftar toko terdekat, sedangkan pada menu akun, *user* dapat mengubah profil atau mengubah password. Tampilan halaman menu utama dapat dilihat pada Gambar 4.6.

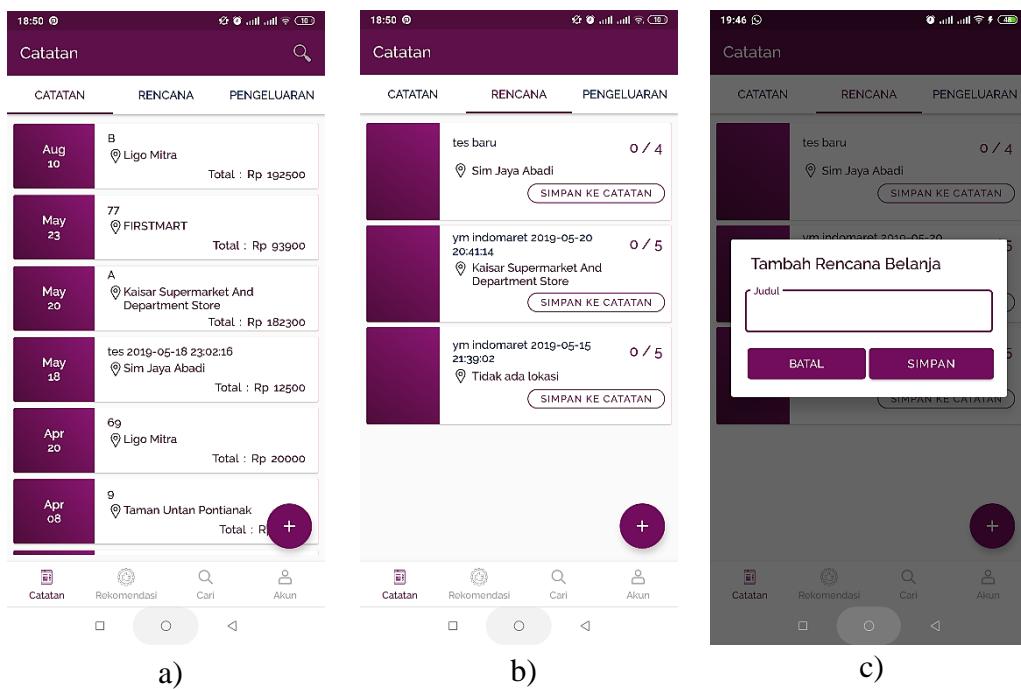


Gambar 4.6 Antarmuka Menu Utama

4.1.9 Tampilan Antarmuka Halaman Catatan

Pada halaman catatan, terdapat 3 menu pada *tab layout* yaitu catatan, rencana belanja, dan pengeluaran. Pada menu catatan, aplikasi menampilkan daftar catatan belanja yang ditambahkan *user* seperti yang terlihat pada gambar a dan pada menu rencana belanja *user* dapat melihat daftar rencana belanjanya yang terlihat pada gambar b. *User* dapat mengubah catatan atau rencana belanja dengan menahan salah satu catatan atau rencana belanja yang akan diubah dan kemudian memilih pilihan edit. Selain itu untuk menghapus catatan atau rencana belanja, *user* dapat melakukannya dengan cara menggeser salah satu catatan atau rencana belanja.

Pada halaman rencana belanja yang dapat dilihat pada gambar b, *user* dapat menambahkan rencana belanja dengan menekan tombol tambah dan kemudian akan aplikasi akan menampilkan *dialog* untuk menambahkan rencana belanja seperti yang terlihat pada gambar c. Halaman catatan dapat digunakan tanpa koneksi internet karena data yang ditampilkan merupakan data yang tersimpan pada penyimpanan lokal. Tampilan halaman catatan dapat dilihat pada Gambar 4.7.



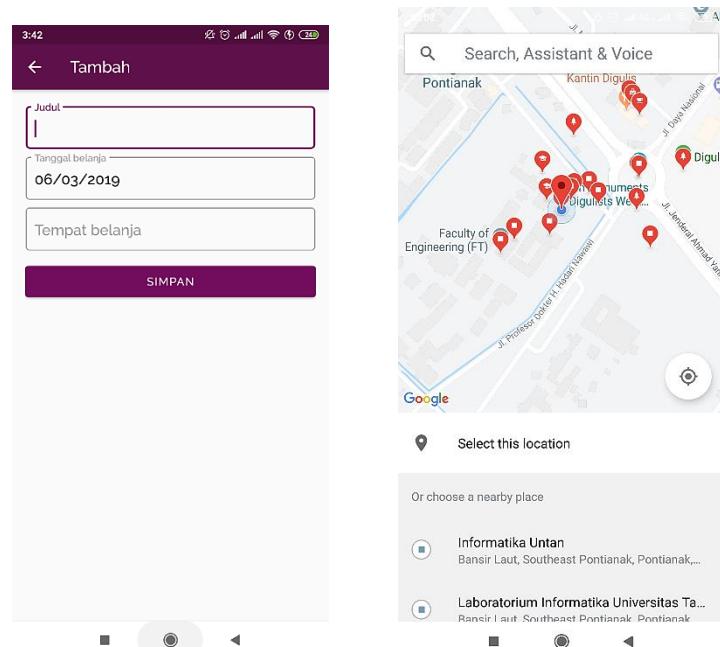
Gambar 4.7 Antarmuka Halaman Catatan

4.1.10 Tampilan Antarmuka Halaman Tambah Catatan Belanja

Pada halaman tambah catatan belanja, *user* diharuskan mengisi judul catatan, tanggal belanja untuk dapat menambahkan catatan belanja. *User* dapat mengosongkan atau mengisi tempat belanja. Pada saat mengisi tempat belanja,

aplikasi akan menampilkan halaman *place picker* dari *Places SDK* untuk Android seperti yang terlihat pada gambar b. Setelah *user* memilih tempat, aplikasi akan kembali ke halaman tambah catatan yang ada pada gambar a dan tempat belanja kemudian akan langsung terisi secara otomatis.

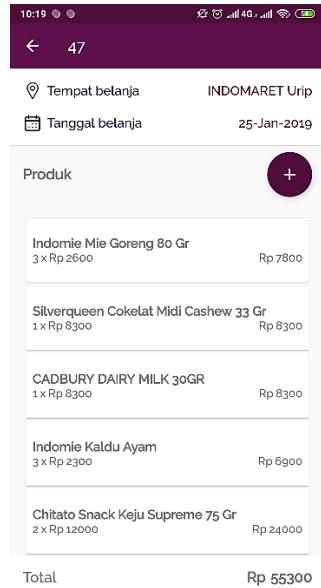
User dapat menambahkan catatan belanja tanpa terkoneksi internet dengan mengosongkan tempat belanja disebabkan oleh data tempat yang diambil oleh *place picker* untuk ditampilkan ke dalam peta merupakan data *online* sehingga membutuhkan koneksi internet. Tampilan halaman tambah catatan belanja dapat dilihat pada Gambar 4.8.



Gambar 4.8 Antarmuka Halaman Tambah Catatan Belanja

4.1.11 Tampilan Antarmuka Halaman Detail Catatan

Pada halaman detail catatan, aplikasi menampilkan detail catatan yaitu tempat belanja, tanggal belanja, dan daftar item belanja. Pada halaman ini *user* juga dapat menambahkan item belanja dengan menekan tombol tambah yang ada pada tampilan aplikasi dan kemudian aplikasi akan menampilkan halaman tambah item catatan dalam bentuk *dialog*. Tampilan halaman detail catatan dapat dilihat pada Gambar 4.9.

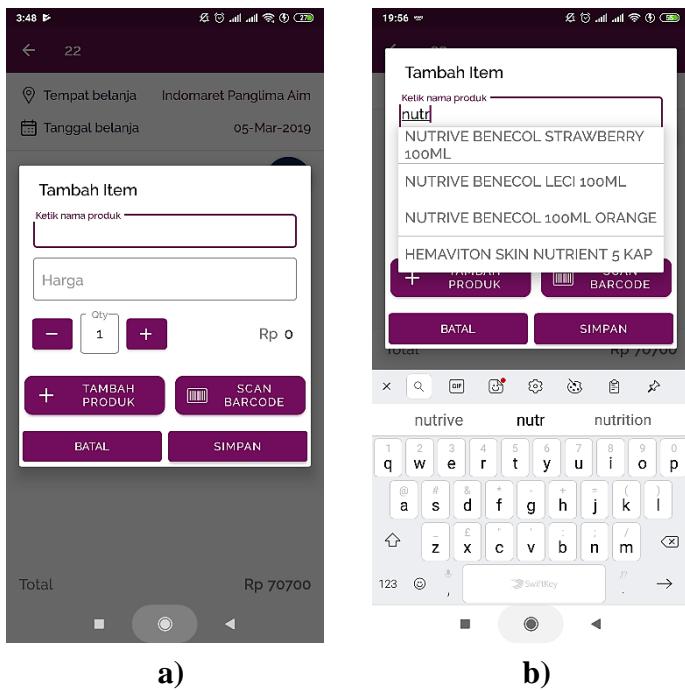


Gambar 4.9 Antarmuka Halaman Detail Catatan

4.1.12 Tampilan Antarmuka Halaman Tambah Item Catatan

Halaman tambah item catatan yang terlihat pada gambar a merupakan halaman dalam bentuk dialog untuk menambahkan item catatan yang dapat diakses melalui tombol tambah pada halaman detail catatan. Pada nama produk diimplementasikan dengan menggunakan komponen *AutocompleteTextView* untuk dapat memberikan *suggestion list* produk kepada *user* ketika *user* mengetik nama produk. *User* dapat mengisi nama produk dengan mengetik nama produk yang diinginkan dan kemudian memilih salah satu produk yang ada pada *suggestion list* seperti yang terlihat pada gambar b. Selain itu *user* juga dapat mengisi nama produk dengan cara *scan barcode* melalui tombol *scan barcode* dan nama produk akan terisi secara otomatis jika produk ditemukan.

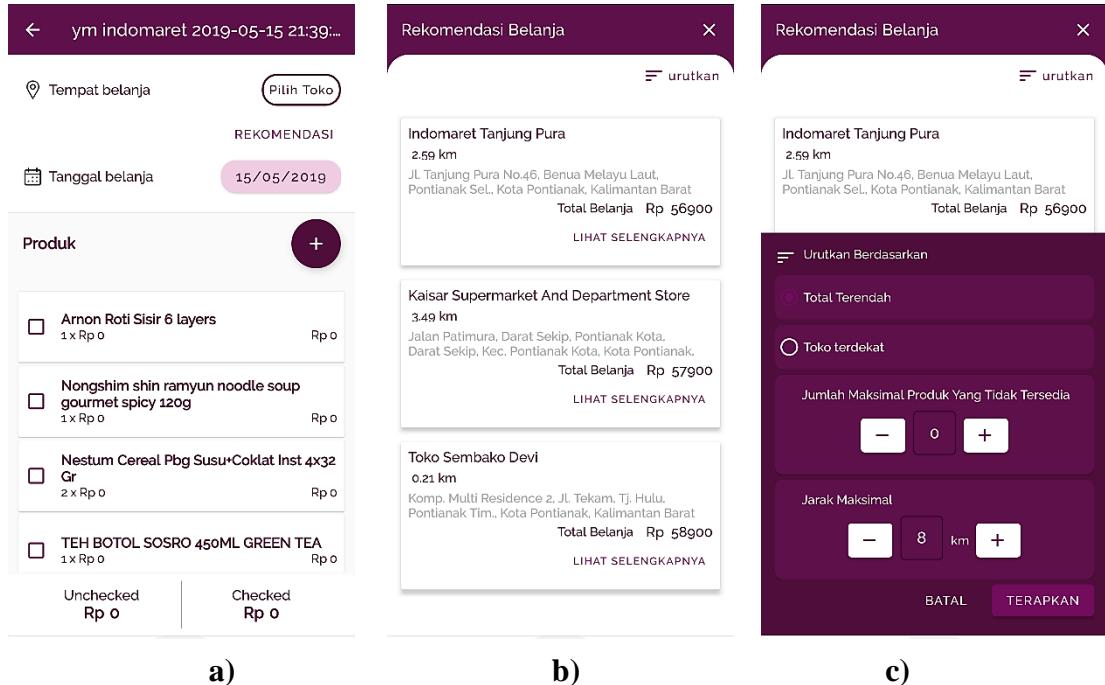
Produk yang tidak ditemukan pada basis data dapat *user* tambahkan pada halaman tambah produk yang dapat diakses melalui tombol tambah produk pada halaman ini. Setelah berhasil memasukkan nama produk *user* diharuskan mengisi harga dan jumlah dan kemudian total akan langsung terhitung untuk kemudian disimpan dengan menekan tombol simpan. Tampilan halaman tambah item catatan dapat dilihat pada Gambar 4.10.



Gambar 4.10 Antarmuka Halaman Tambah Item Catatan

4.1.13 Tampilan Antarmuka Halaman Detail Rencana Belanja

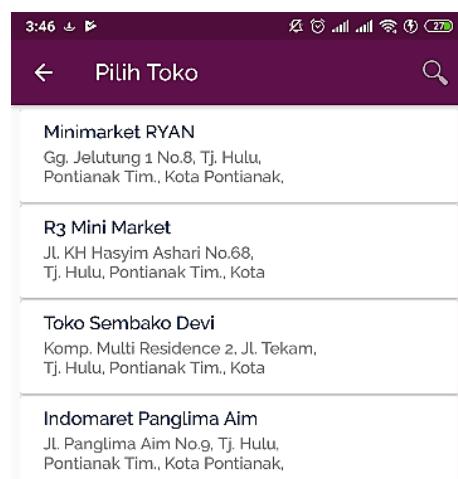
Pada halaman detail rencana belanja, *user* dapat mengisi detail belanja dengan memilih tanggal belanja, memilih toko, atau menambahkan item rencana belanja terlebih dahulu. Pada saat *user* menambahkan item rencana, jika toko belum dipilih maka harga pada daftar akan terisi nol dan kemudian harga akan terupdate saat *user* telah memilih toko seperti yang terlihat pada gambar a. Setelah *user* menambahkan item rencana belanja, *user* dapat melihat rekomendasi tempat belanja dengan total belanja terendah seperti yang terlihat pada gambar b dan *user* dapat mengatur urutan rekomendasi berdasarkan total terendah atau toko terdekat serta menentukan jarak maksimal toko yang ditampilkan dan jumlah maksimal produk tidak tersedia yang diperbolehkan pada rekomendasi. Tampilan halaman detail rencana belanja dapat dilihat pada Gambar 4.11.



Gambar 4.11 Antarmuka Halaman Detail Rencana Belanja

4.1.14 Tampilan Antarmuka Halaman Pilih Toko

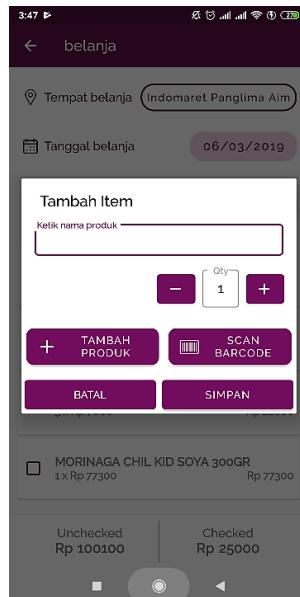
Halaman pilih toko merupakan halaman yang ditampilkan untuk memilih toko pada rencana belanja dan pilih toko pada menu bandingkan harga. Toko yang ditampilkan dalam daftar merupakan daftar toko yang pernah dimasukkan oleh pengguna aplikasi sebagai tempat belanjanya pada catatan belanja. Toko yang ditampilkan merupakan daftar toko yang diurutkan berdasarkan jarak dari lokasi *user*. Tampilan halaman pilih toko dapat dilihat pada Gambar 4.12.



Gambar 4.12 Antarmuka Halaman Pilih Toko

4.1.15 Tampilan Antarmuka Halaman Tambah Item Rencana

Pada halaman tambah item rencana, *user* dapat menambahkan item dengan memasukkan nama produk dan jumlah produk yang akan dibeli. Pada saat memasukkan nama produk, seperti pada halaman tambah item catatan belanja, *user* dapat memasukkan produk dengan mengetik nama atau *scan barcode* produk. Selain itu *user* juga dapat menambahkan produk baru jika produk yang diinginkan belum tersedia pada basis data dengan menekan tombol tambah produk. Tampilan halaman tambah item rencana dapat dilihat pada Gambar 4.13.



Gambar 4.13 Antarmuka Halaman Tambah Item Rencana

4.1.16 Tampilan Antarmuka Halaman Tambah Produk

Pada halaman tambah produk, *user* dapat menambahkan produk baru yang belum tersedia pada basis data dengan memasukkan *barcode* dan nama produk. Produk yang dapat ditambahkan hanya produk yang memiliki barcode. *User* dapat memasukkan nomor barcode dengan menggunakan *barcode scanner* yang ada tombol disamping kanan atas. Tampilan halaman tambah produk dapat dilihat pada Gambar 4.14.



Gambar 4.14 Antarmuka Halaman Tambah Produk

4.1.17 Tampilan Antarmuka Halaman Rekomendasi

Pada halaman rekomendasi, terdapat informasi lokasi kota *user* berada. *User* dapat mengakses rekomendasi dengan menekan tombol rekomendasi seperti yang terlihat pada gambar a dan kemudian memilih rencana belanja yang akan dilihat rekomendasinya. Kemudian aplikasi akan menampilkan daftar hasil rekomendasi seperti yang terlihat pada gambar b dan *user* dapat mengubah urutan hasil rekomendasi berdasarkan toko terdekat atau total terendah dan menentukan jarak maksimal toko yang dihasilkan pada rekomendasi. Selain itu, *user* juga dapat melihat detail rekomendasi berupa daftar produk yang tersedia dan tidak tersedia beserta harga dari masing-masing produk. Antarmuka halaman rekomendasi dapat dilihat pada Gambar 4.15.

a)

Produk	Toko	Jarak	Total Belanja
MORINAGA CHIL KID SOYA 300GR	FIRSTMART	0.46 km	Rp 84000
Nestum Cereal Pbg Susu+Coklat Inst 4x32 Gr	FIRSTMART	0.46 km	Rp 9900
Yum Yum Tom Yum Udang 70Gr	Ligo Mitra	0.46 km	Rp 5000
Teh Pucuk Teh MN Pucuk Harum Btl 500ML		0.21 km	

b)

Produk	Toko	Jarak	Total Belanja
Sim Jaya Abadi	Sim Jaya Abadi	0.58 km	Rp 56900
Indomaret Tanjung Pura	Indomaret Tanjung Pura	2.59 km	Rp 56900
Kaisar Supermarket And Department Store	Kaisar Supermarket And Department Store	3.49 km	Rp 57900
Toko Sembako Devi	Toko Sembako Devi	0.21 km	

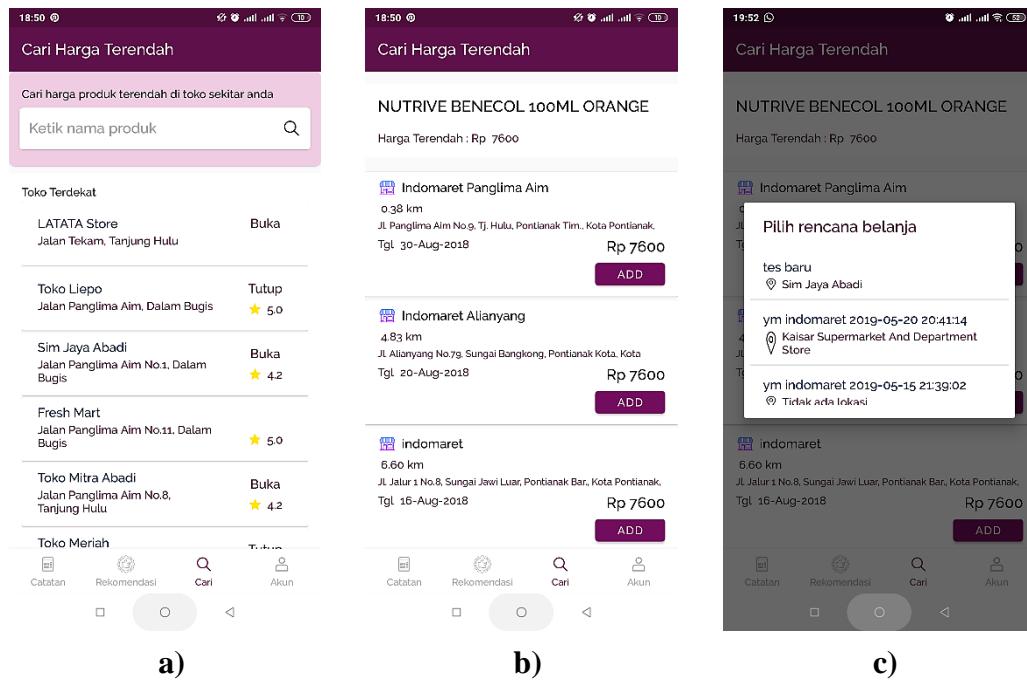
c)

Total	Rp 56900
-------	----------

Gambar 4.15 Antarmuka Halaman Rekomendasi

4.1.18 Tampilan Antarmuka Halaman Cari

Pada halaman cari, *user* dapat melihat harga dengan mengetik nama produk yang akan dilihat terlebih dahulu seperti yang terlihat pada gambar a, kemudian aplikasi akan menampilkan daftar harga terbaru dari berbagai toko dan diurutkan berdasarkan toko yang memiliki harga terendah dan *user* dapat memilih atau mencari toko seperti yang terlihat pada gambar a, kemudian melihat produk yang tersedia pada toko yang dapat dilihat pada gambar b. Selain itu pada halaman cari, *user* dapat melihat daftar toko terdekat seperti terlihat pada gambar a. Daftar produk yang tersedia pada toko dapat dilihat dengan memilih salah satu toko. Pada saat melihat produk toko, *user* dapat menambahkan produk pada rencana belanja. Daftar produk yang ditampilkan merupakan data yang dihasilkan dari catatan belanja *user*. Pada halaman cari harga terendah seperti yang terlihat pada gambar b, *user* dapat menambahkan produk tersebut pada rencana belanja yang sudah dibuat oleh *user* seperti yang terlihat pada gambar c. Tampilan halaman cari harga terendah dapat dilihat pada Gambar 4.16.

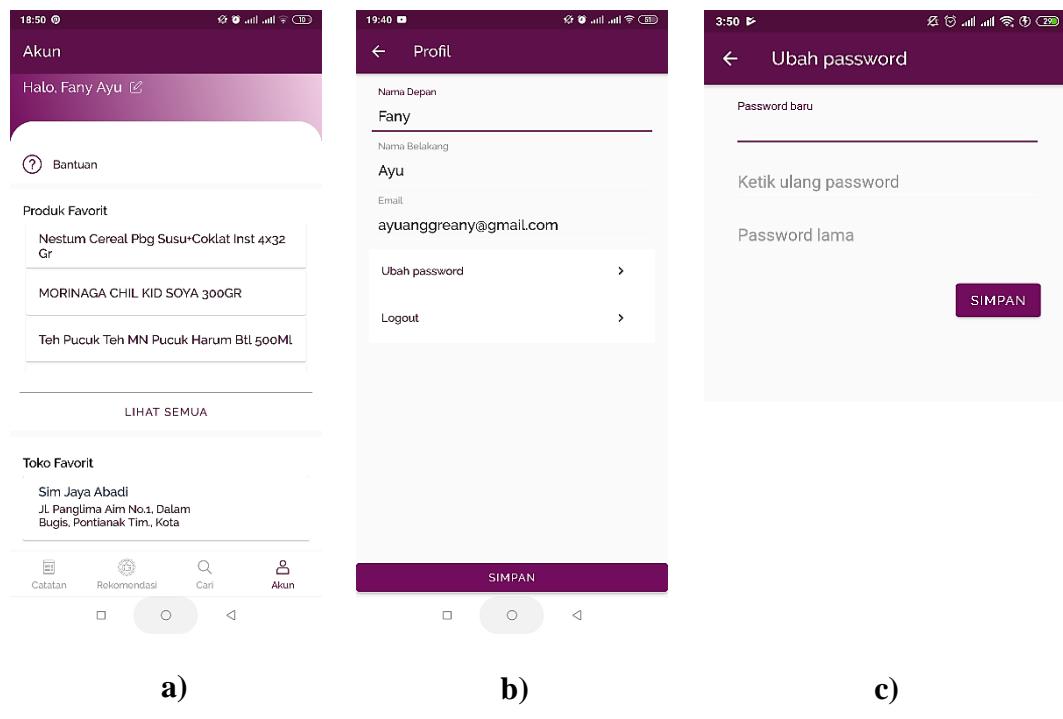


Gambar 4.16 Antarmuka Halaman Cari Harga Terendah

4.1.19 Tampilan Antarmuka Halaman Akun

Pada halaman akun seperti terlihat pada gambar a, *user* dapat melihat bantuan, produk favorit, dan toko favorit. Produk favorit dan toko favorit dihasilkan

dari catatan belanja *user*. *User* dapat mengubah profil, ubah *password*, dan *logout* dengan menekan *icon* yang terletak disebelah kanan nama *user* seperti yang terlihat pada gambar a. Pada saat *user* memilih ubah profil maka aplikasi akan menampilkan halaman untuk mengubah profil seperti terlihat pada gambar b dan saat *user* memilih ubah *password*, aplikasi akan menampilkan halaman untuk ubah password seperti terlihat pada gambar c. Tampilan halaman akun dapat dilihat pada Gambar 4.17.



Gambar 4.17 Antarmuka Halaman Akun

4.1.20 Tampilan Antarmuka Halaman Bantuan

Pada halaman bantuan, terdapat informasi versi aplikasi dan informasi penggunaan aplikasi yang dapat diakses oleh user dengan memilih menu akun pada *bottom navigation view* dan menekan tombol bantuan. Antarmuka halaman bantuan dapat dilihat pada Gambar 4.18.



Gambar 4.18 Antarmuka Halaman Bantuan

4.1.21 Tampilan Antarmuka Halaman Pengeluaran

Halaman pengeluaran dapat diakses oleh pengguna dengan memilih menu akun pada *bottom navigation view*. Pada halaman pengeluaran, *user* dapat melihat total belanja perbulan seperti yang terlihat pada gambar a, perminggu pada gambar b, dan perhari pada gambar c. Data yang ditampilkan merupakan perhitungan dari catatan belanja yang sudah *user* masukkan. Pada halaman perbulan, *user* dapat melihat total belanjanya pada bulan tertentu yang dikelompokkan menjadi total belanja pertoko. Antarmuka halaman pengeluaran dapat dilihat pada Gambar 4.19.

<p>a)</p>	<p>b)</p>	<p>c)</p>
------------------	------------------	------------------

Gambar 4.19 Antarmuka Halaman Pengeluaran

4.2 Hasil Pengujian

Aplikasi catatan belanja menggunakan dua pengujian yang pertama pengujian BlackBox dan yang kedua pengujian Kuesioner.

4.2.1 Hasil Pengujian Harga Terendah

Pengujian dilakukan dengan menggunakan data pada tabel riwayat harga yang terdapat pada basis data. Berikut ini digunakan data produk Ever E Vit-E 250 dengan id produk 1106, terdapat 3 data harga dengan 2 harga pada toko yang sama dan satu harga pada toko berbeda. Pada 2 data harga di toko yang sama yaitu Indomaret Merdeka, terdapat harga produk 14900 pada tanggal 24 januari 2019 dan harga produk 15900 pada 11 februari 2019 sehingga data yang seharusnya ditampilkan yaitu harga produk 15900 pada 11 februari 2019. Berdasarkan 3 data berikut, data yang seharusnya ditampilkan pada aplikasi yaitu harga produk 15900 pada tanggal 25 februari 2019 di toko Johar 10 Indomaret dan harga produk 15900 pada tanggal 11 februari 2019 di toko Indomaret Merdeka. Daftar harga uji dapat dilihat pada gambar 4.20.

+ Options						
id_produk	nama_produk	tanggal_belanja	harga	nama_toko	alamat	
1106	Ever E Vit-E 250 Vegicaps Strip 6 Pcs	2019-01-24	14900	Indomaret Merdeka	Jl. Merdeka Barat No.633B, Mariana, Pontianak Kota...	
1106	Ever E Vit-E 250 Vegicaps Strip 6 Pcs	2019-02-25	15900	Johar 10 Indomaret	Jl. Johar No.23, Darat Sekip, Pontianak Kota, Kota...	
1106	Ever E Vit-E 250 Vegicaps Strip 6 Pcs	2019-02-11	15900	Indomaret Merdeka	Jl. Merdeka Barat No.633B, Mariana, Pontianak Kota...	

Gambar 4.20 Data uji harga terendah Ever E Vit-E 250

. Hasil pengujian menampilkan harga terendah produk Ever E Vit E 250 yaitu 15900 pada toko Johar 10 Indomaret dan 15900 pada Indomaret Merdeka sesuai dengan yang diharapkan. Berikut ini merupakan hasil harga terendah pada aplikasi terlihat pada gambar 4.21.

Cari Harga Terendah

Ever E Vit-E 250 Vegicaps Strip 6 Pcs

Harga Terendah : Rp 15900

Johar 10 Indomaret

4.42 km

JL. Johar No.23, Darat Sekip, Pontianak Kota, Kota Pontianak.

Tgl 25-Feb-2019

Rp 15900

ADD

Indomaret Merdeka

4.69 km

Jl. Merdeka Barat No.633B, Mariana, Pontianak Kota, Kota

Tgl 11-Feb-2019

Rp 15900

ADD

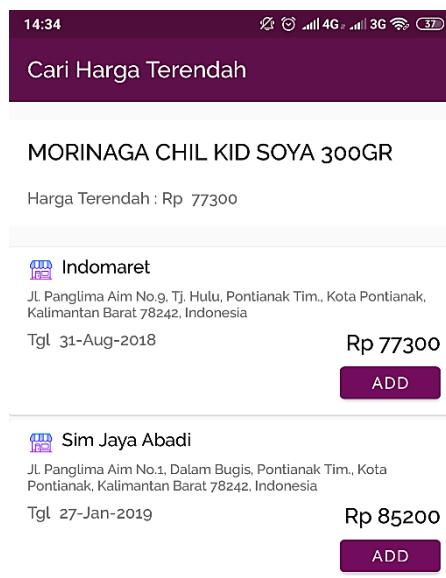
Gambar 4.21 Hasil pengujian harga terendah Ever E Vit-E 250 pada aplikasi

Berikut ini digunakan data produk Morinaga Chil Kid Soya 300gr dengan id produk 1028, terdapat 7 data harga dengan harga yang sama pada setiap toko dan tanggal belanja yang berbeda. Data yang seharusnya ditampilkan yaitu harga dengan tanggal belanja terbaru yaitu tanggal 31 agustus 2018 pada toko Indomaret dan 27 januari 2019 pada toko Sim Jaya Abadi. Daftar harga uji dapat dilihat pada gambar 4.22.

+ Options						
id_produk	nama_produk	tanggal_belanja	harga	nama_toko	alamat	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-29	77300	Indomaret	Jl. Panglima Aim No.9, Tj. Hulu, Pontianak Tim., K...	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-31	77300	Indomaret	Jl. Panglima Aim No.9, Tj. Hulu, Pontianak Tim., K...	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-21	77300	Indomaret	Jl. Panglima Aim No.9, Tj. Hulu, Pontianak Tim., K...	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-16	77300	Indomaret	Jl. Panglima Aim No.9, Tj. Hulu, Pontianak Tim., K...	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-14	85200	Sim Jaya Abadi	Jl. Panglima Aim No.1, Dalam Bugis, Pontianak Tim...	
1028	MORINAGA CHIL KID SOYA 300GR	2018-08-11	85200	Sim Jaya Abadi	Jl. Panglima Aim No.1, Dalam Bugis, Pontianak Tim...	
1028	MORINAGA CHIL KID SOYA 300GR	2019-01-27	85200	Sim Jaya Abadi	Jl. Panglima Aim No.1, Dalam Bugis, Pontianak Tim...	

Gambar 4.22 Data uji harga terendah Morinaga Chil Kid Soya 300gr

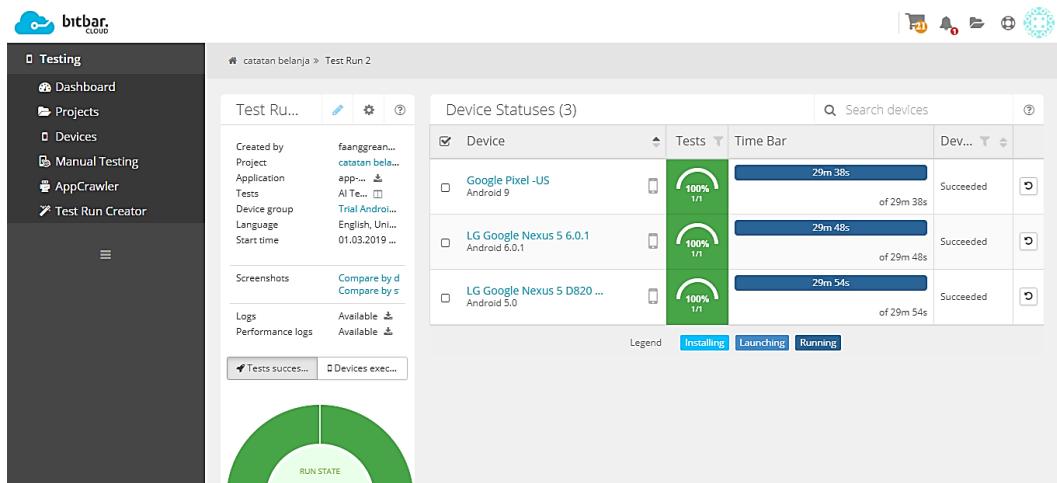
Hasil pengujian menampilkan harga terendah produk Morinaga Chil Kid Soya yaitu 77300 dengan tanggal 31 agustus 2018 pada toko indomaret dan 85200 dengan tanggal 27 januari 2019 pada toko sim jaya abadi sesuai dengan yang diharapkan. Berikut ini merupakan hasil harga terendah pada aplikasi terlihat pada gambar 4.23.



Gambar 4.23 Hasil pengujian harga terendah Morinaga Chil Kid Soya 300gr pada aplikasi

4.2.2 Hasil Pengujian Alpha

Pengujian dilakukan untuk mengetes fungsionalitas, performa, dan memori yang terdapat dalam penelitian yang dilakukan. Alat yang digunakan adalah Bitbar Testing dengan menggunakan AI Testbot yang merupakan pengujian otomatis berbasis *Artificial Intelligent*, berikut adalah tahapan pengujian yang telah dilakukan.

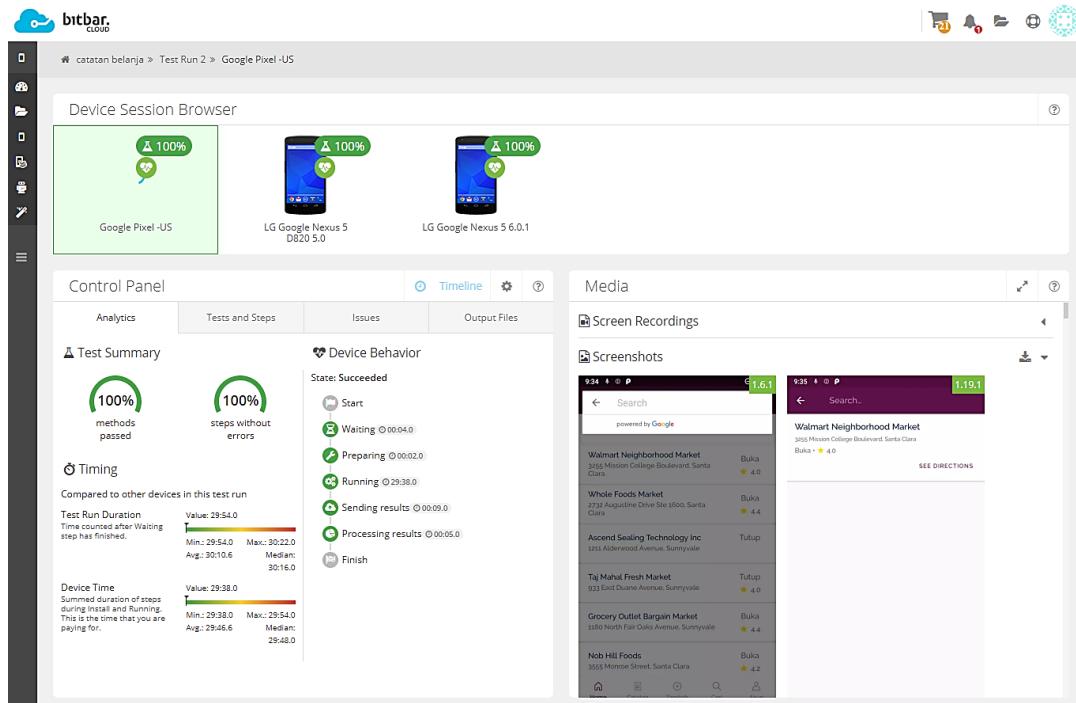


Gambar 4.24 Pengujian alpha berhasil

Pengujian dilakukan dalam tiga perangkat yaitu Google Pixel-US, LG Google Nexus 5 6.0.1, dan LG Google Nexus 5 D820. Bagian yang diujikan adalah fungsional, performa, dan memori, berikut adalah penjabaran hasil pengujian. Pengujian pada Google Pixel US

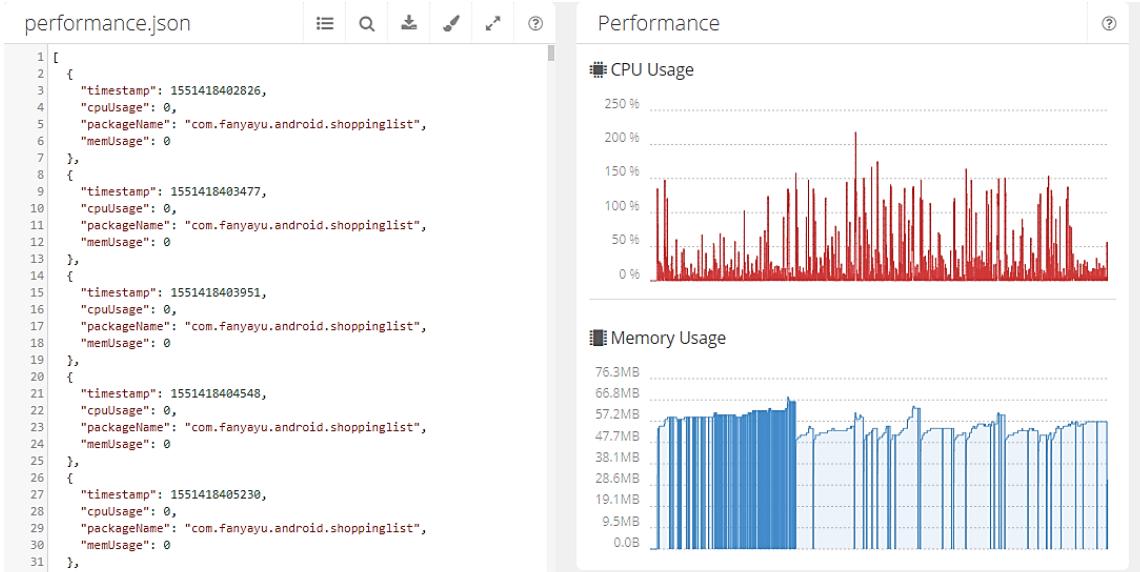
4.2.2.1 Pengujian pada Google Pixel US

Pengujian pada perangkat Google Pixel US dengan android versi 9.0 yang telah dilakukan melalui Bitbar Testing. Pada halaman awal yaitu analisis dari pengujian metode didapatkan hasil 100%, menyatakan bahwa seluruh metode di dalam aplikasi dapat berjalan. Selanjutnya yaitu pengujian tahapan tanpa mengalami *error* didapatkan hasil pengujian yaitu 100% yang menunjukkan seluruh tahapan yang dilakukan aplikasi berjalan tanpa gangguan. Berikut merupakan gambar hasil analisis pengujian pada gambar 4.25.



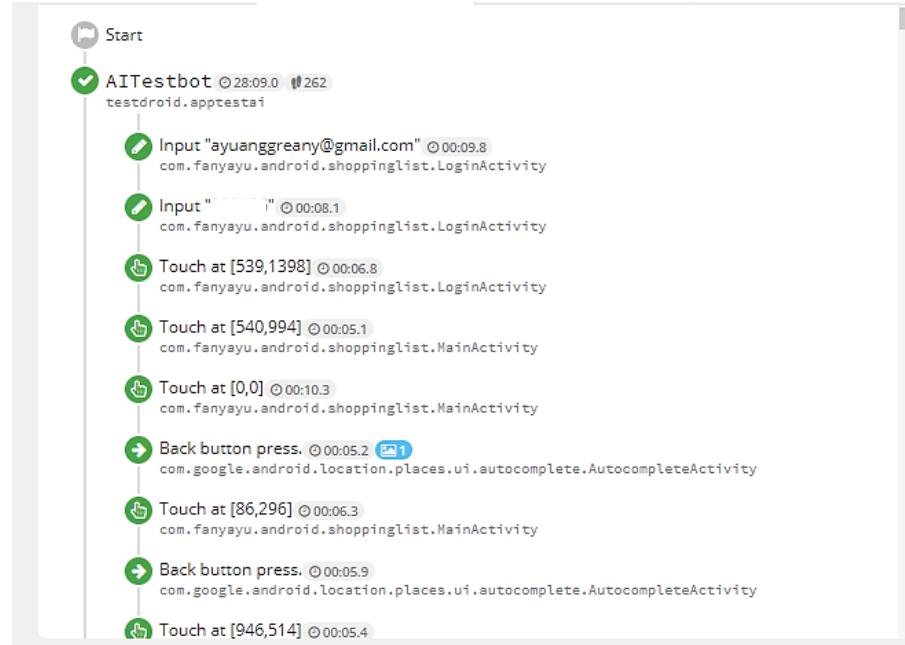
Gambar 4.25 Hasil pengujian analisis pada Google Pixel US

Pengujian performa atau penggunaan CPU yaitu dalam kisaran 11.6% dan penggunaan memori yaitu dalam kisaran 57.2MB. Berdasarkan hasil pengujian penggunaan memori cukup tinggi pada perangkat, dapat dilihat dalam gambar 4.26.



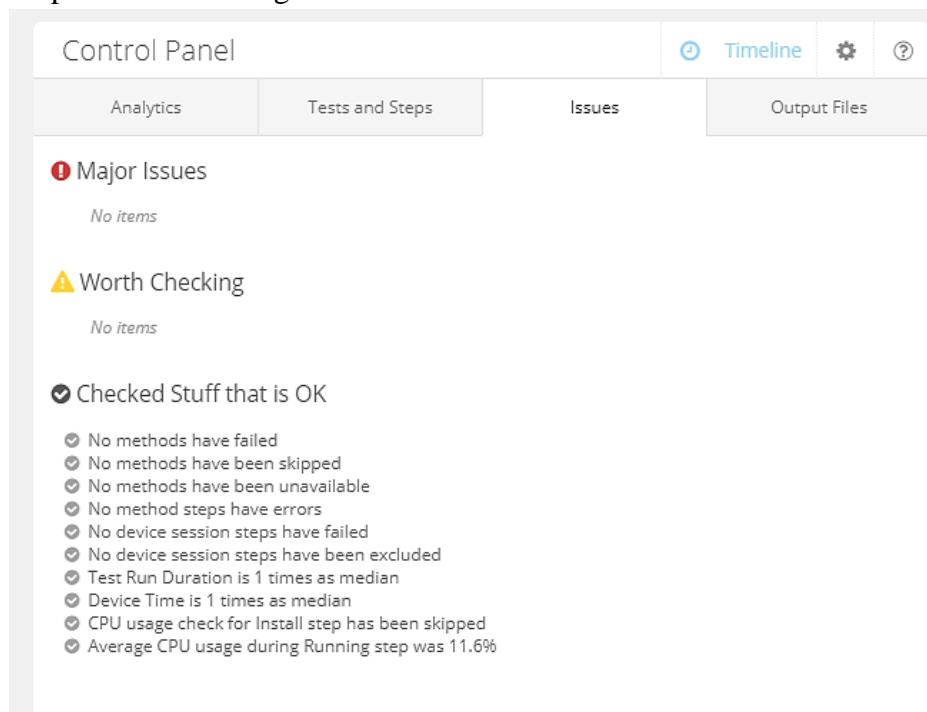
Gambar 4.26 Hasil pengujian performa aplikasi pada Google Pixel US

Langkah pengujian dan dokumentasi dari pengujian dapat dilihat dalam gambar 4.27.



Gambar 4.27 Hasil pengujian langkah pada Google Pixel US

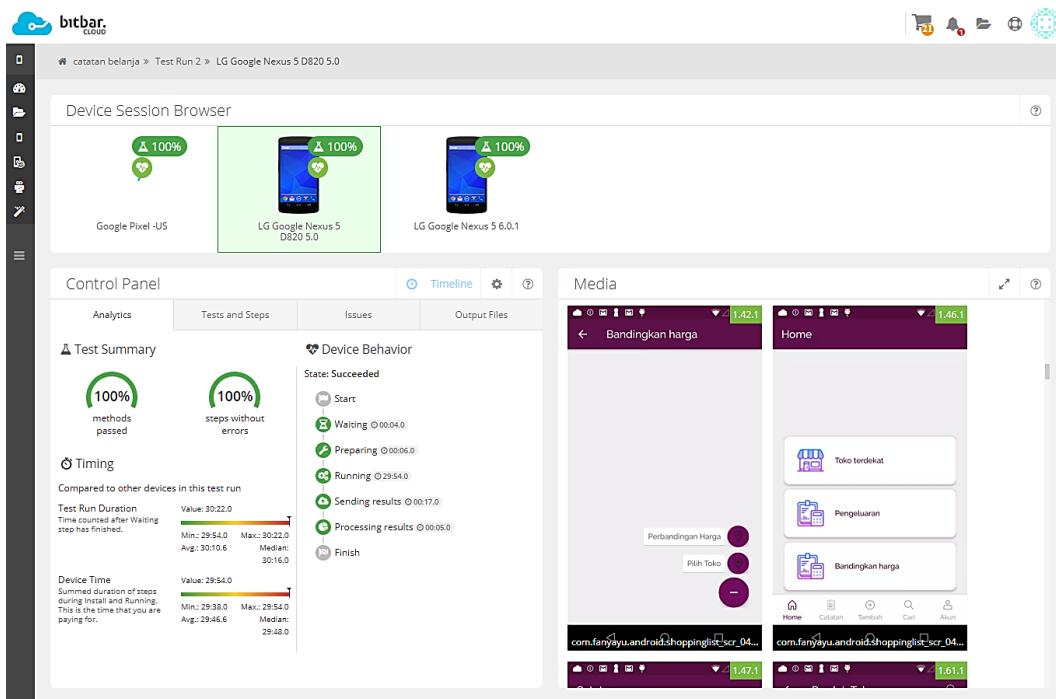
Hasil pengujian memperlihatkan tidak ada permasalahan besar dan tidak ada hal yang sebaiknya diperiksa kembali pada aplikasi. Hasil pengujian masalah pada aplikasi dapat dilihat dalam gambar 4.28.



Gambar 4.28 Hasil pengujian permasalahan aplikasi pada Google Pixel US

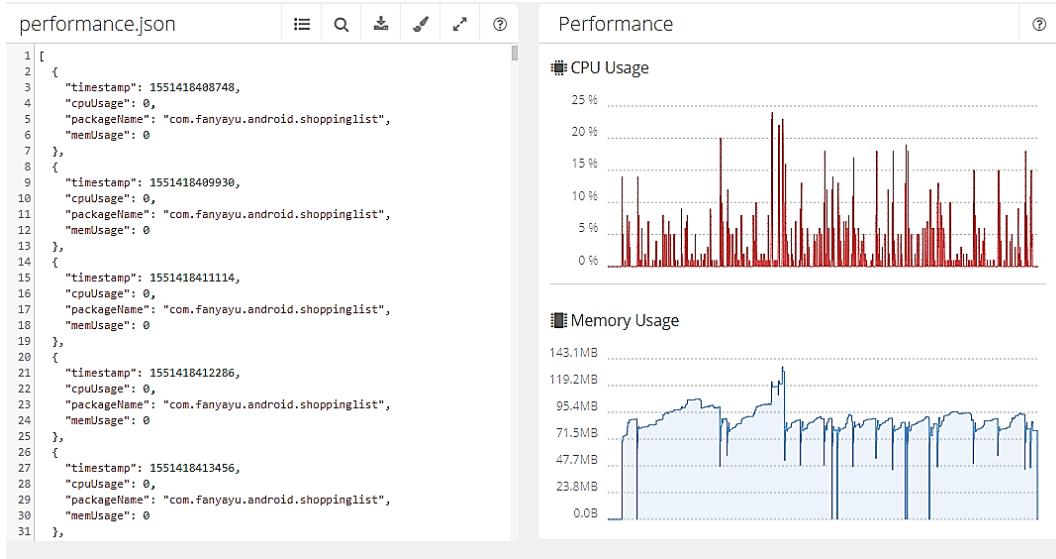
4.2.2.2 Pengujian pada LG Google Nexus 5 D820

Pengujian pada perangkat LG Google Nexus 5 D820 dengan android versi 5.0 yang telah dilakukan melalui Bitbar Testing. Pada halaman awal yaitu analisis dari pengujian metode didapatkan hasil 100%, menyatakan bahwa seluruh metode di dalam aplikasi dapat berjalan. Selanjutnya yaitu pengujian tahapan tanpa mengalami *error* didapatkan hasil pengujian yaitu 100% yang menunjukkan seluruh tahapan yang dilakukan aplikasi berjalan tanpa gangguan. Berikut merupakan gambar hasil analisis pengujian pada gambar 4.29.



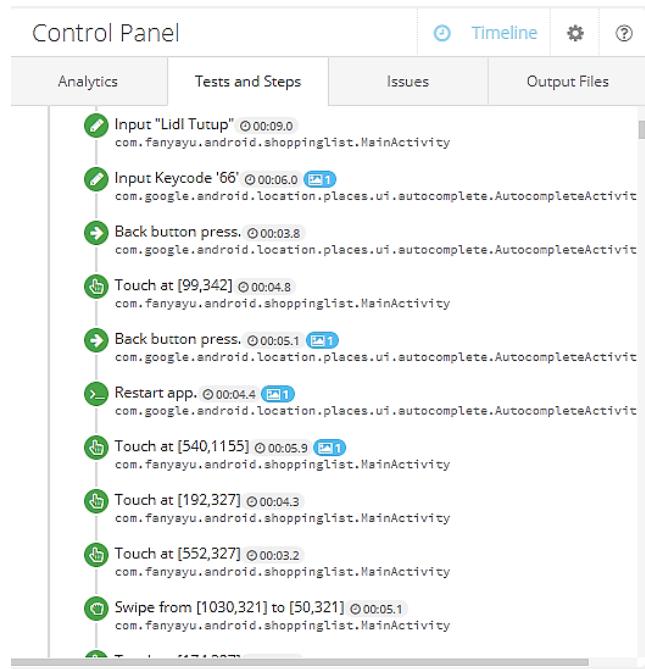
Gambar 4.29 Hasil Pengujian Analisis LG Google Nexus 5 D820

Pengujian performa atau penggunaan CPU yaitu dalam kisaran 8.2% dan penggunaan memori yaitu dalam kisaran 95.4MB. Berdasarkan hasil pengujian penggunaan memori cukup tinggi pada perangkat, dapat dilihat dalam gambar 4.30.



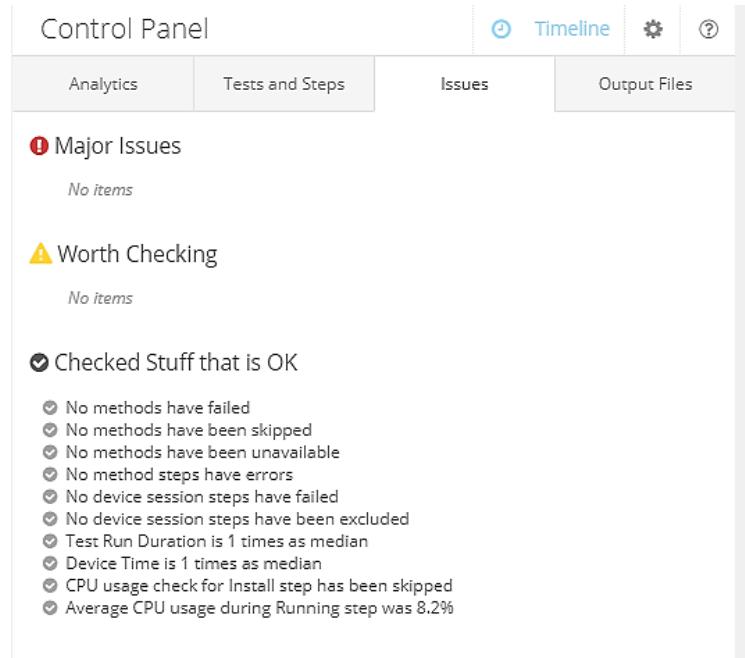
Gambar 4.30 Hasil Pengujian Performa Aplikasi Pada LG Google Nexus 5 D820

Langkah pengujian dan dokumentasi dari pengujian dapat dilihat dalam gambar 4.31.



Gambar 4.31 Hasil Pengujian Langkah LG Google Nexus 5 D820

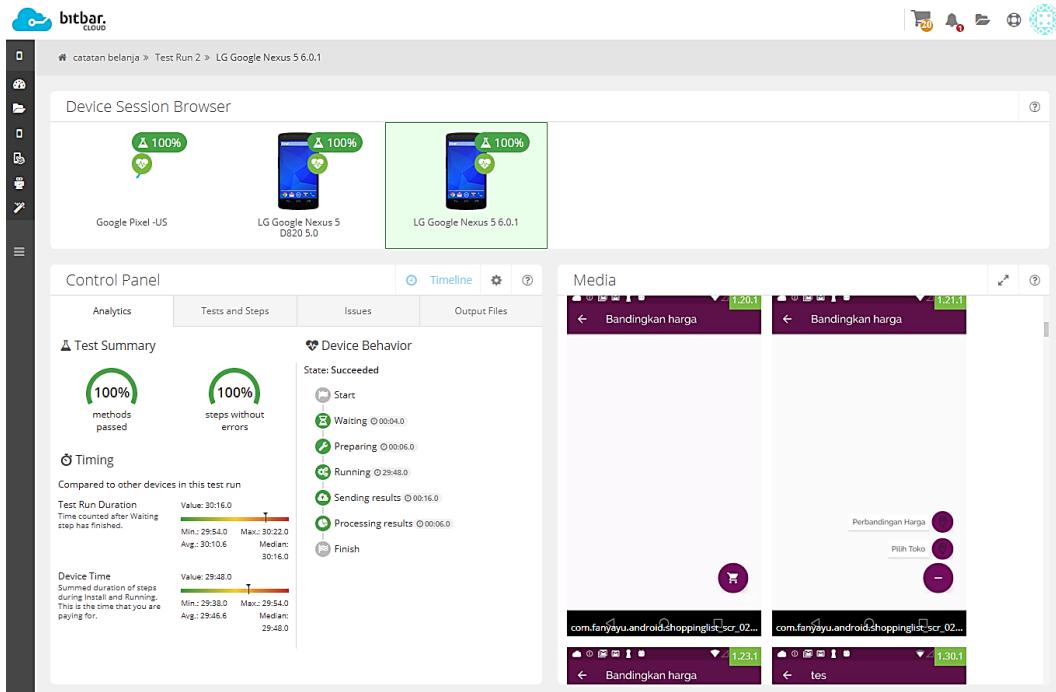
Hasil pengujian memperlihatkan tidak ada permasalahan besar dan tidak ada hal yang sebaiknya diperiksa kembali pada aplikasi. Hasil pengujian masalah pada aplikasi dapat dilihat dalam gambar 4.32.



Gambar 4.32 Hasil Pengujian Permasalahan Aplikasi Pada LG Google Nexus 5 D820

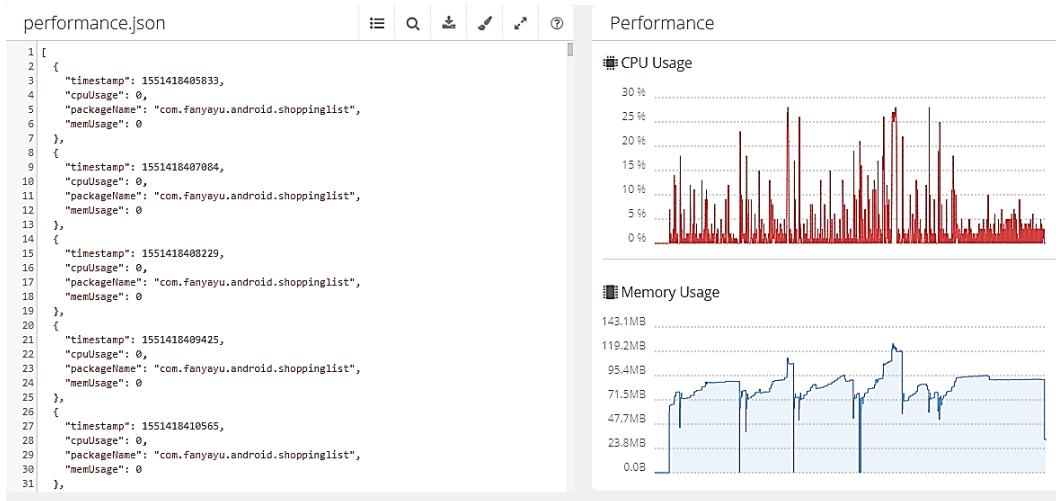
4.2.2.3 Pengujian pada LG Google Nexus 5 6.0.1

Pengujian pada perangkat LG Google Nexus 5 6.0.1 dengan android versi 6.0.1 yang telah dilakukan melalui Bitbar Testing. Pada halaman awal yaitu analisis dari pengujian metode didapatkan hasil 100%, menyatakan bahwa seluruh metode di dalam aplikasi dapat berjalan. Selanjutnya yaitu pengujian tahapan tanpa mengalami *error* didapatkan hasil pengujian yaitu 100% yang menunjukkan seluruh tahapan yang dilakukan aplikasi berjalan tanpa gangguan. Berikut merupakan gambar hasil analisis pengujian pada gambar 4.33.



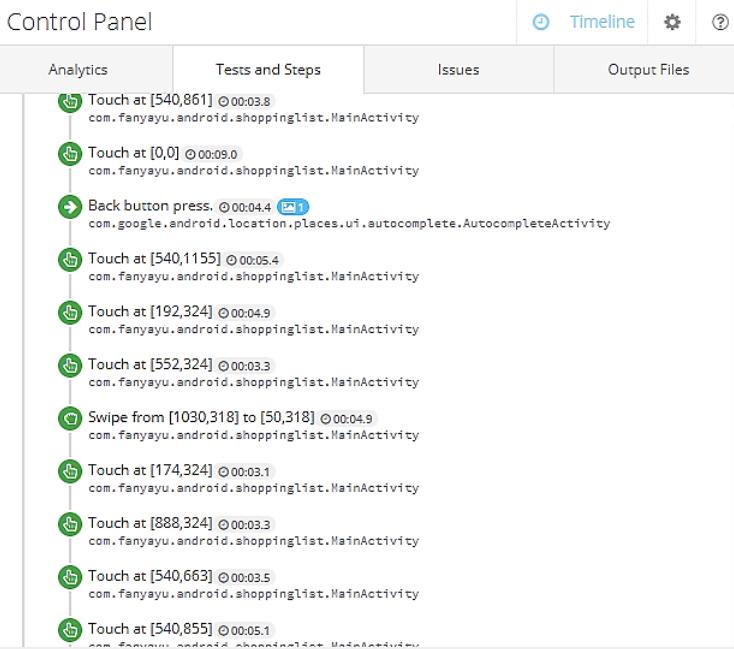
Gambar 4.33 Hasil Pengujian Analisis LG Google Nexus 5 6.0.1

Pengujian performa atau penggunaan CPU yaitu dalam kisaran 6.6% dan penggunaan memori yaitu dalam kisaran 90.4MB. Berdasarkan hasil pengujian penggunaan memori cukup tinggi pada perangkat, dapat dilihat dalam gambar 4.34.



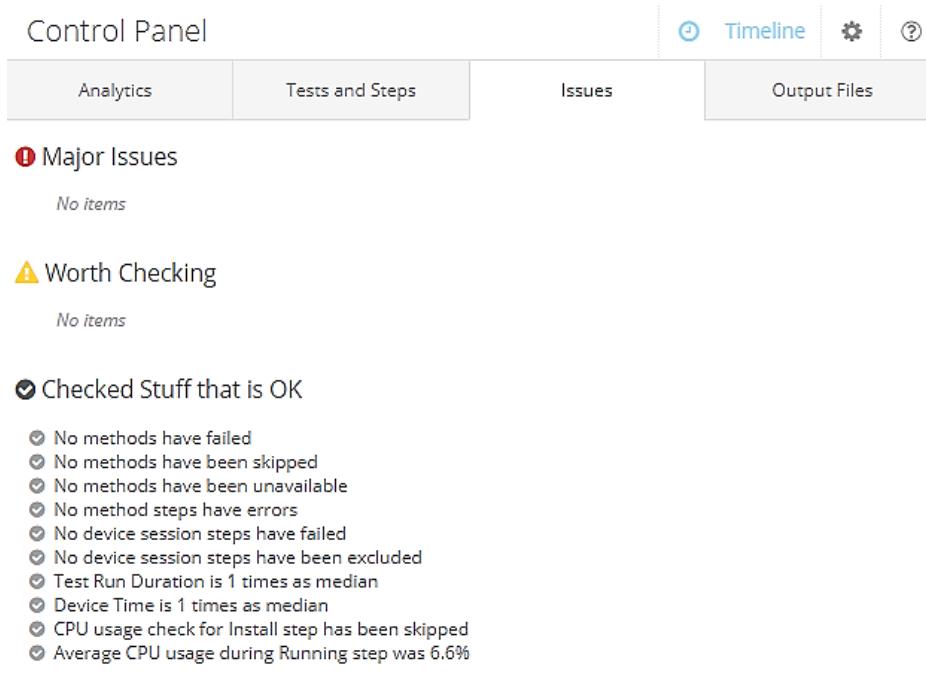
Gambar 4.34 Hasil Pengujian Performa LG Google Nexus 5 6.0.1

Langkah pengujian dan dokumentasi dari pengujian dapat dilihat dalam gambar 4.35.



Gambar 4.35 Hasil Pengujian Langkah LG Google Nexus 5 6.0.1

Hasil pengujian memperlihatkan tidak ada permasalahan besar dan tidak ada hal yang sebaiknya diperiksa kembali pada aplikasi. Hasil pengujian masalah pada aplikasi dapat dilihat dalam gambar 4.36.



Gambar 4.36 Hasil Pengujian Permasalahan Aplikasi Pada LG Google Nexus 5 6.0.1

4.2.3 Hasil Pengujian Jarak Terdekat

Pengujian dilakukan dengan membandingkan antara jarak hasil perhitungan formula haversine dengan jarak hasil perhitungan google maps untuk mendapatkan hasil yang efektif. Perbandingan dilakukan dengan menggunakan koordinat default yaitu Gedung Informatika Untan, Pontianak.

Tabel 4.1 Hasil perbandingan perhitungan Formula Haversine dengan Google Maps

No	Nama Toko	Jarak Haversine (Km)	Jarak Google Maps (Km)	Selisih
1	Taman Untan Pontianak	0.33	0.05	0.28
2	Hypermart Mega Mall Pontianak	0.39	0.55	0.16
3	Alma Convenience Store	0.77	0.63	0.14
4	Indomaret Sepakat II	0.78	0.59	0.19
5	Indomaret Tanjung Pura	1.97	2.00	0.03
6	Ligo Mitra	2.45	2.56	0.11
7	Indomaret Sungai Raya Dalam ll/ A2	2.68	2.56	0.12
8	Kaisar Supermarket And Department Store	3.12	3.23	0.11
9	Sim Jaya Abadi	3.44	3.29	0.15
10	Johar 10 Indomaret	3.57	3.75	0.18
11	INDOMARET Urip	3.61	3.74	0.13
12	Indomaret Alianyang	3.66	3.87	0.21
13	Indomaret Panglima Aim	3.58	3.44	0.14
14	Indomaret Merdeka	4.15	4.33	0.18
15	Alfamart Gusti Situt Mahmud	4.67	4.61	0.06

Dari pengujian yang telah dilakukan didapatkan hasilnya yang dapat dilihat pada tabel 4.1 bahwa sistem rekomendasi harga produk terendah yang dihasilkan

sistem menghasilkan selisih yang kecil dan tidak berbeda jauh dengan hasil perhitungan jarak dari Google Maps dengan perhitungan dengan menggunakan Formula Haversine.

4.2.4 Hasil Pengujian Kuesioner

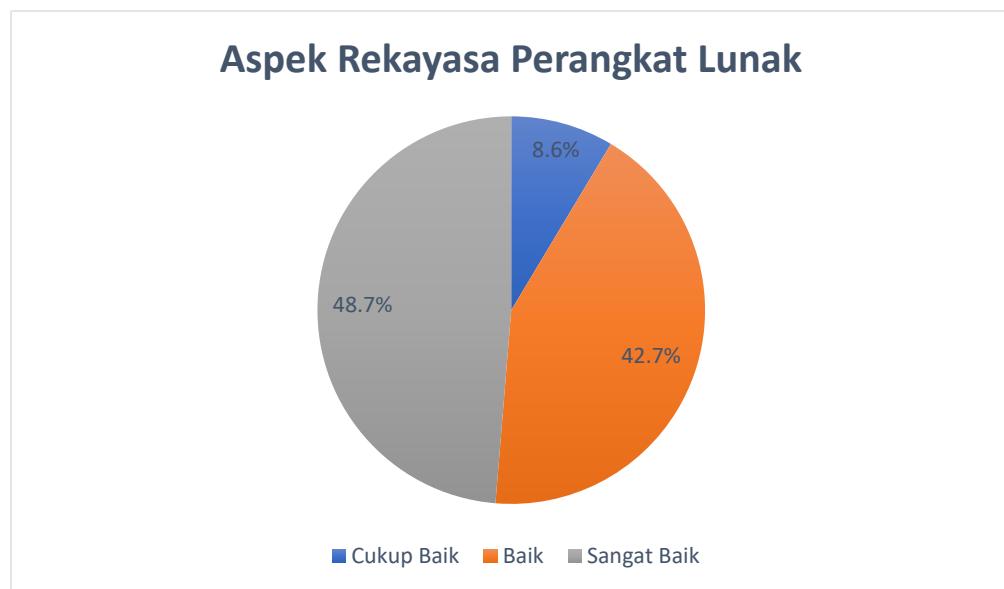
Kuesioner yang dibagikan terdiri dari beberapa pertanyaan yang dikelompokkan menjadi 3 aspek, yaitu aspek rekayasa perangkat lunak, aspek fungsionalitas, dan aspek komunikasi visual. Kuesioner dibagikan kepada 30 responden yang merupakan pengguna Android. Untuk penilaian kuesioner diberikan nilai 1 untuk tidak baik, 2 untuk kurang baik, 3 untuk cukup baik, 4 untuk baik, dan 5 untuk sangat baik.

Hasil kuesioner terhadap aspek rekayasa perangkat lunak ditunjukkan pada Tabel 4.2.

Tabel 4.2 Hasil Kuesioner Aspek Rekayasa Perangkat Lunak Pada Aplikasi

No	Aspek Rekayasa Perangkat Lunak	Tanggapan					Total
		1	2	3	4	5	
1	Kemudahan menjalankan aplikasi	0	0	2	11	17	30
2	Kompatibilitas aplikasi pada perangkat	0	0	2	17	11	30
3	Kelancaran menjalankan aplikasi pada perangkat	0	0	1	11	18	30
4	Kemudahan mengakses fitur – fitur pada aplikasi	0	0	3	13	14	30
5	Kenyamanan dalam penggunaan aplikasi secara keseluruhan	0	0	5	12	13	30
Jumlah		0	0	13	64	73	150
Percentase (%)		0	0	8,6	42,7	48,7	100

Berdasarkan hasil kuesioner pada aspek rekayasa perangkat lunak pada aplikasi diketahui bahwa responden menanggapi dengan sangat baik. Hasil pengujian menunjukkan presentase cukup baik sebesar 8,6%, baik dengan presentase 42,7%, dan sangat baik 48,7%.

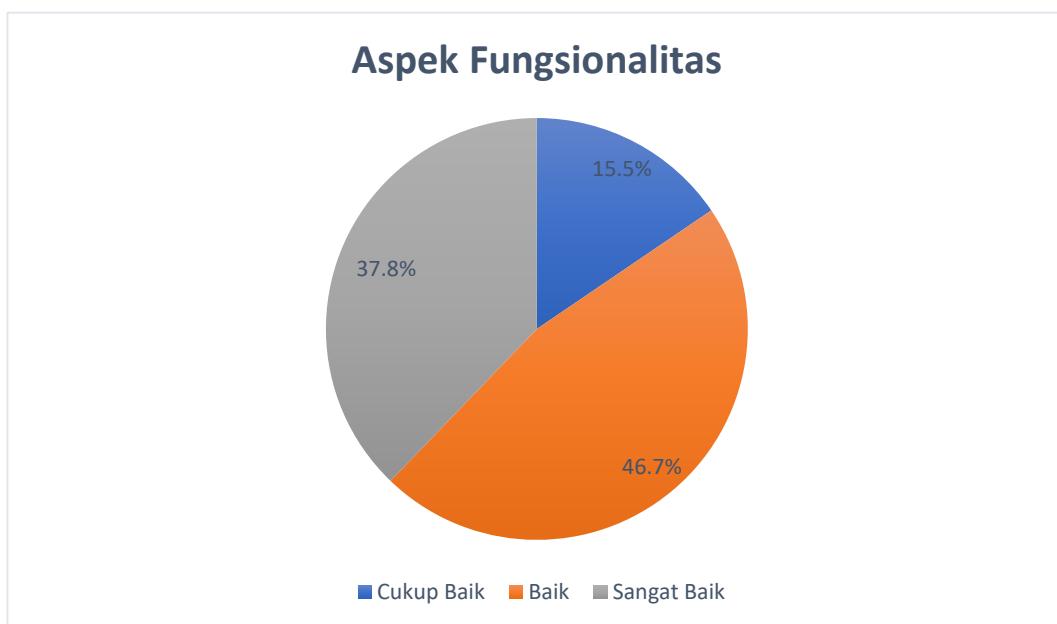


Gambar 4.37 Chart Hasil Pengujian Aspek Rekayasa Perangkat Lunak

Tabel 4.3 Hasil kuesioner aspek fungsionalitas

No	Aspek Fungsionalitas	Tanggapan					Total
		1	2	3	4	5	
1	Kinerja aplikasi saat menampilkan data	0	0	6	14	10	30
2	Tingkat kesesuaian menampilkan hasil harga produk terendah	0	0	4	16	10	30
3	Kinerja aplikasi saat melakukan manajemen data	0	0	4	12	14	30
Jumlah		0	0	14	42	34	90
Percentase (%)		0	0	15,5	46,7	37,8	100

Berdasarkan hasil kuesioner pada aspek fungsional pada aplikasi dapat diketahui responden menanggapi dengan baik. Hasil pengujian menunjukkan presentase cukup baik sebesar 15,5%, baik sebesar 46,7%, dan sangat baik 37,8%.

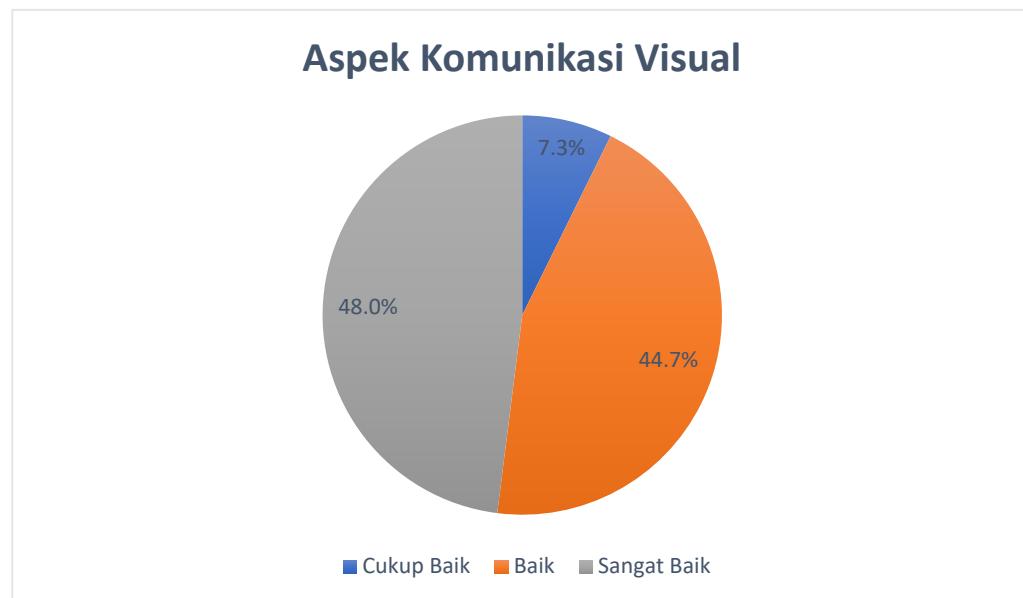


Gambar 4.38 Chart Hasil Pengujian Aspek Fungsionalitas

Tabel 4.4 Hasil kuesioner aspek komunikasi visual

No	Aspek Komunikasi Visual	Tanggapan					Total
		1	2	3	4	5	
1	Tampilan (antarmuka) aplikasi	0	0	2	11	17	30
2	Jenis dan ukuran huruf yang digunakan mudah dibaca	0	0	0	15	15	30
3	Kombinasi warna pada tampilan aplikasi	0	0	3	14	13	30
4	Respon (<i>feedback</i>) aplikasi terhadap input yang dimasukkan	0	0	3	16	11	30
5	Kemudahan memahami informasi yang ditampilkan pada aplikasi	0	0	3	11	16	30
Jumlah		0	0	11	67	72	150
Persentase (%)		0	0	7,3	44,7	48	100

Berdasarkan hasil kuesioner pada aspek komunikasi visual aplikasi dapat diketahui bahwa responden menanggapi dengan baik. Hasil pengujian menunjukkan presentase cukup baik sebesar 7,3%, baik sebesar 44,7%, dan sangat baik sebesar 48%.



Gambar 4.39 Chart Hasil Pengujian Aspek Komunikasi Visual

4.2.5 Likert's Summated Rating (LSR)

Untuk melihat skor terbesar dan terkecil dari satu orang responden dan total semua responden dari kuesioner ditunjukkan pada Tabel 4.4.

Tabel 4.5 Total Skor Responden dari Kuesioner

Responden	Item													Total
	1	2	3	4	5	6	7	8	9	10	11	12	13	
1	5	5	5	5	5	5	5	5	5	5	5	5	5	65
2	5	5	5	5	5	5	5	5	5	5	5	5	5	65
3	5	5	5	5	5	5	5	5	5	5	5	5	5	65
4	4	4	4	4	4	4	4	4	4	4	4	4	4	52
5	5	4	5	5	4	4	4	5	4	5	3	4	4	56
6	5	4	4	5	4	4	4	5	3	5	4	4	5	56
7	4	4	5	4	3	4	3	4	4	4	4	3	4	50
8	5	5	5	5	5	5	5	5	5	5	5	5	5	65
9	5	5	5	5	5	5	5	5	5	5	5	5	5	65
10	4	4	4	3	4	3	3	3	4	5	3	3	4	47

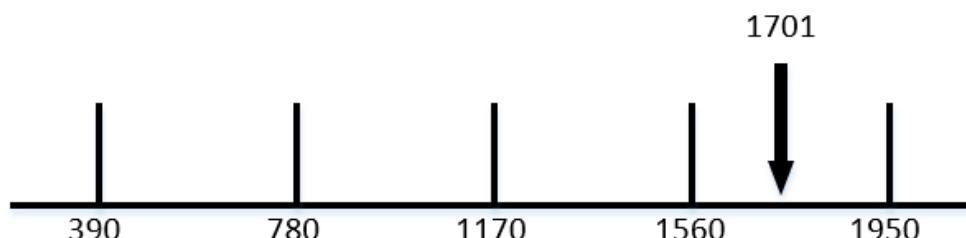
Responden	Item													Total
	1	2	3	4	5	6	7	8	9	10	11	12	13	
11	3	4	4	4	4	4	4	4	4	4	4	4	4	51
12	5	5	5	4	4	5	4	5	5	5	5	5	5	62
13	3	4	3	4	4	4	4	4	4	4	4	4	4	50
14	4	4	5	5	5	5	4	5	5	4	4	5	5	60
15	5	4	4	5	3	3	4	3	4	4	4	5	4	52
16	5	4	5	5	5	5	5	4	5	5	5	4	5	62
17	4	3	4	4	5	3	4	3	5	4	4	4	5	52
18	5	4	4	4	5	4	5	4	4	4	4	4	4	55
19	4	4	4	3	3	3	3	3	4	4	4	4	3	46
20	4	4	5	4	3	4	4	4	3	4	4	3	3	49
21	4	5	5	5	3	3	4	5	5	4	5	5	5	58
22	5	5	5	4	5	4	5	4	5	5	4	4	5	60
23	4	3	5	3	4	3	3	4	5	4	5	4	3	50
24	5	5	5	4	5	5	5	4	5	5	5	4	5	62
25	5	5	5	5	5	5	5	5	5	5	5	5	5	65
26	4	4	5	4	5	4	4	5	4	5	3	4	5	56
27	5	4	4	5	4	4	4	5	5	5	5	4	5	59
28	5	4	4	5	4	4	4	5	4	4	5	5	4	57
29	4	4	4	4	4	4	4	4	5	4	4	4	4	53
30	5	5	5	4	4	4	4	4	5	4	4	4	4	56
Total Skor														1701

Data yang diperoleh dari hasil pengujian dengan kuesioner kemudian diukur dengan metode *Likert's Summated Rating* (LSR).

1. Jumlah skor untuk setiap responden:

- Skor maksimal = 65 (5 x 13 item)
- Skor minimal = 13 (1 x 13 item)
- Skor median = 39 (3 x 13 item)
- Skor kuartil I = 26 (2 x 13 item)
- Skor kuartil III = 52 (4 x 13 item)

2. Jumlah skor untuk seluruh responden:
 - Maksimal = 1950 (30×65)
 - Minimal = 390 (30×13)
 - Median = 1170 (30×39)
 - Kuartil I = 780 (30×26)
 - Kuartil II = 1560 (30×52)
3. Interpretasi jumlah skor :
 - $1560 < \text{skor} < 1950$, artinya aplikasi dinilai sangat positif (program dinilai berhasil)
 - $1170 < \text{skor} < 1560$, artinya aplikasi dinilai positif (program dinilai cukup berhasil)
 - $780 < \text{skor} < 1170$, artinya aplikasi dinilai negatif (program dinilai kurang berhasil)
 - $390 < \text{skor} < 780$, artinya aplikasi dinilai sangat negatif (program dinilai tidak berhasil)



Gambar 4.40 Hasil Kuesioner Pada Interpretasi *Likert's Summated Rating* (LSR)

Pada Gambar 4.37 menunjukkan bahwa hasil penelitian berada di antara skor 1560 dan 1950 yang artinya, hasil kuesioner menandakan responden menilai aplikasi sangat positif dan dinilai berhasil.

4.3 Analisis Hasil Pengujian

Rincian hasil analisis pengujian aplikasi catatan belanja yang telah dilakukan adalah sebagai berikut:

1. Hasil pengujian fungsional yang dilakukan dengan menggunakan sistem oleh perusahaan Bitbar menunjukkan semua metode yang ada dalam aplikasi dapat berjalan dengan lancar.

2. Hasil pengujian memori dan *cpu usage* yang dilakukan oleh sistem dari perusahaan Bitbar menunjukkan aplikasi catatan belanja menggunakan banyak memori untuk menjalankan aplikasi.
3. Hasil pengujian jarak terdekat yang dilakukan dengan membandingkan hasil perhitungan formula haversine dan hasil perhitungan jarak Google Maps API menunjukkan bahwa sistem menghasilkan akurasi yang cukup baik dalam menentukan jarak terdekat antara pengguna dan toko.
4. Hasil pengujian harga terendah menunjukkan sistem dapat menampilkan harga produk terendah yang merupakan harga terbaru yang ada pada basis data.
5. Berdasarkan hasil kuesioner, secara umum responden menerima dengan baik aplikasi catatan belanja dalam penelitian yang dilakukan.
6. Hasil pengujian kuesioner yang diukur dengan metode *Likert's Summated Rating* (LSR) menunjukkan responden menilai aplikasi yang dibagun dengan “sangat baik” dengan skor 1701.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil analisis dan pengujian terhadap aplikasi catatan belanja untuk menghasilkan rekomendasi harga produk terendah, maka dapat ditarik kesimpulan sebagai berikut:

1. Pendekatan partisipasi masyarakat digunakan untuk mendapatkan data produk yang terbaru berdasarkan catatan belanja tiap pengguna pada aplikasi catatan belanja yang telah dirancang dan dibuat di penelitian ini.
2. Berdasarkan hasil pengujian, aplikasi dapat digunakan untuk melakukan pencatatan belanja dan melihat rekomendasi harga terendah produk dari berbagai toko terdekat. Sistem dapat menghasilkan rekomendasi harga produk terendah yang sesuai dan merupakan harga terbaru yang ada pada basis data. Dari pengujian jarak terdekat, sistem menghasilkan jarak dengan akurasi yang cukup baik dalam menentukan jarak antara pengguna dan toko yang digunakan pada rekomendasi.
3. Hasil kuesioner yang dihitung dengan metode *Likert's Summated Rating* menunjukkan responden menilai aplikasi yang dibagun dengan “sangat baik” dengan skor 1701.

5.2 Saran

Adapun beberapa hal yang perlu ditambahkan dalam pengembangan aplikasi ini, adalah sebagai berikut:

1. Penambahan fitur untuk scan struk belanja sehingga memudahkan pengguna dalam menambahkan item catatan belanja.
2. Navigasi menu pada *user interface* aplikasi dibuat dengan menggunakan pedoman desain Material yang merupakan sistem pedoman, komponen, dan alat yang mendukung untuk mengembangkan desain antarmuka yang baik sehingga dapat menghasilkan aplikasi yang mudah digunakan oleh *user*.
3. Penambahan fitur untuk pemesanan produk secara *online* melalui rencana belanja pada aplikasi sehingga *user* dapat dengan mudah berbelanja.
4. Penambahan sistem saldo yang mandiri untuk melakukan pembayaran

- pemesanan produk secara *online*.
5. Mengurangi atau meminimalisir penggunaan cpu dan memori pada aplikasi catatan belanja, agar aplikasi menjadi lebih ringan dan optimal di perangkat Android smartphone.