BAB 2
TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Tahap tinjauan pustaka pada perusahaan ini merupakan peninjauan terhadap tempat penelitian studi kasus yang dilakukan di PT. XYZ.

2.1.1 Sejarah PT. XYZ

PT. XYZ didirikan pada tanggal 16 Agustus 1972 berlokasi di Jl. Lengkong Besar No. 11 Bandung dengan nama "CV. XYZ". Nama "XYZ" merupakan singkatan dari Lilin, Pelumas, Rapid dan Gas yang merupakan produk-produk Pertamina saat itu.

Pada tahun 1977 perusahaan menempati lokasi baru di Jl. Emong No. 21 Bandung, yang sampai sekarang menjadi Kantor Pusat dan *Showroom* penjualan kompor dan minyak pelumas Pertamina. Selain memasarkan gas elpiji dan minyak pelumas, produk Pertamina tersedia juga produk-produk kompor dan *sparepart*-nya, *water heater* dan *rice cooker.*

Saat ini PT. XYZ telah memiliki cabang yang tersebar di kota Bandung dan Cimahi, yaitu:

Utara	:	Jl. Sederhana No. 26, Bandung.
		Telp. 2030230 – 2033030
Selatan	:	Jl. Kopo No. 537, Bandung.
		Telp. 5402902 – 2410300
		Jl. Terusahan Buah Batu No. 284, Bandung.
		Telp. 7500300 – 7501313
Cimahi	:	Jl. Encep Kartawiria No. 7, Citeureup, Cimahi
		Telp. 6644650 – 6642732
Barat	:	Jl. Rajawali Barat No. 233, Bandung
		Telp. 6012017 – 6016987

Timur : Jl. Jend. Ahmad Yani No. 624, Bandung

Telp. 7204400- 7272176

Jl. Gatot Subroto No. 229, Bandung

Telp. 7304028 - 7319625

2.1.2 Visi dan Misi

2.1.2.1 Visi

Melayani pengiriman gas elpiji untuk rumah tangga dan industri. Ketepatan dan kecepatan waktu pengiriman konsumen selalu menjadi prioritas kami.

2.1.2.2 Misi

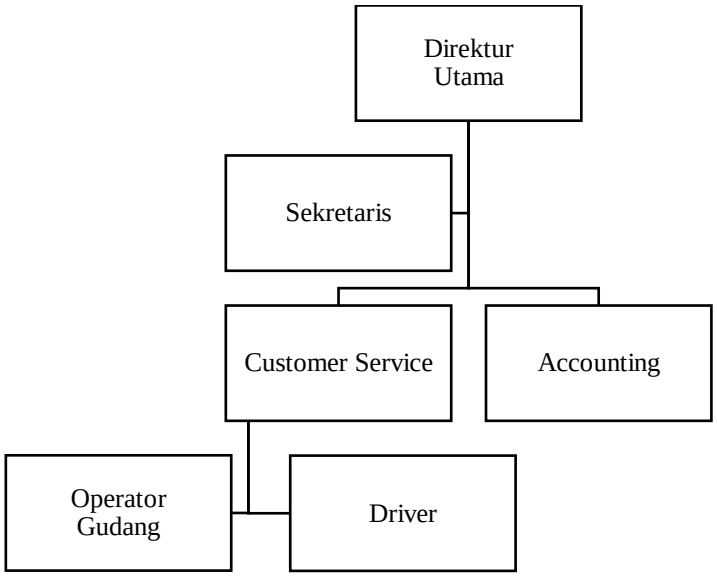
Menyediakan kebutuhan gas dan pelumas selalu tersedia, dan memberi pelayanan terbaik.

2.1.3 Logo Perusahaan

Gambar 2.1 Logo PT. XYZ

Logo merupakan gambar yang memiliki arti tertentu biasanya mewakili suatu arti dari perusahaan ataupun organisasi tersebut, dimana logo juga memudahkan masyrakat untuk mengingat sebuah organisasi ataupun perusahaan serta menjadi pembeda bagi organisasi ataupun perusahaan lainnya. Begitupun PT. XYZ memiliki sebuah logo yang dapat dilihat pada gambar 2.1.

2.1.4 Struktur Organisasi PT. XYZ



Gambar 2.2 Struktur Organisasi PT. XYZ

2.1.5 Deskripsi Tugas

Dapat dilihat dari gambar 2.2 berikut deskripsi tugas dari setiap jabatan yang ada dalam struktur organisasi PT. XYZ :

1. Direktur utama, bertugas dalam mengambil sebuah keputusan dan melakukan pengawasan dalam pekerjaan.
2. Sekretaris, membantu menyusun jadwal direktur utama, dan menrima laporan (menyusun) sebelum diberi ke Direktur Utama.
3. *Accounting,* bertugas dalam menghitung keseluruhan data keuangan perusahaan. Dan melakukan aktifitas-aktifitas keuangan lainnya, seperti pajak dan lain-lain.
4. *Customer Service*, bertugas menerima panggilan dan pesanan, memberikan faktur kepada *driver* serta membuat laporan transaksi harian untuk diberikan kepada *accounting.*
5. *Driver,* bertugas mengantarkan gas kepada konsumen dan melaporkan uang yang diterima kepada operator.
6. Operator Gudang, bertugas memastikan gudang aman, dan ketersediaan gas.

2.2 Landasan Teori

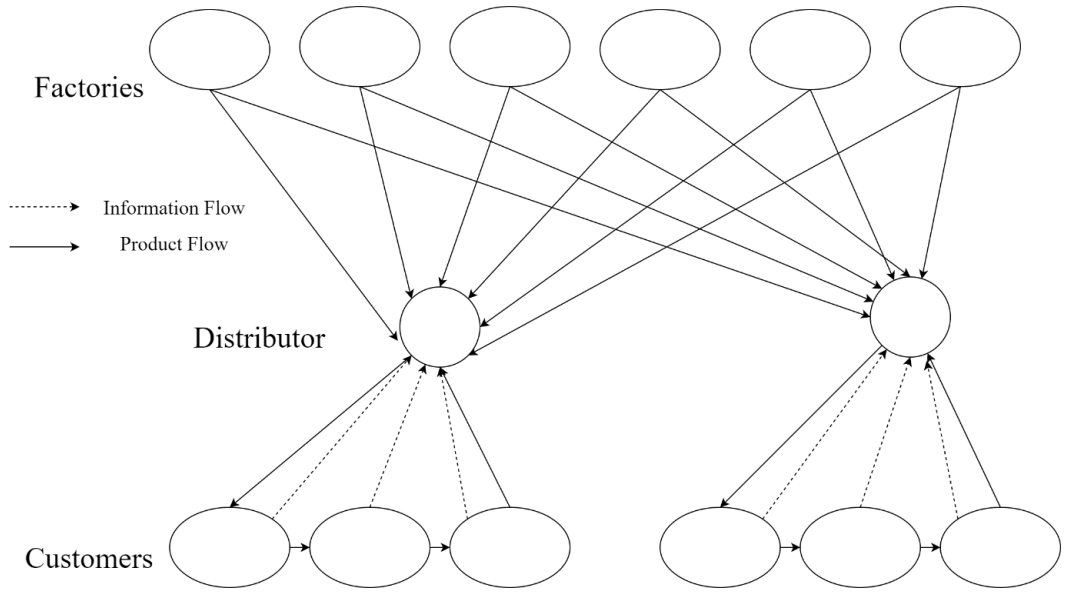
Landasan Teori merupakan definisi dari konsep yang telah disusun secara terstruktur dan dasar yang kuat untuk melandasi sebuah penelitian. Landasan teori yang akan digunakan dalam pembangunan aplikasi Optimalisasi Distribusi Barang PT. XYZ dengan memanfaatkan beberapa teori yang terkait yaitu Model Distribusi Barang, Android, *Weighted Product*, *Global Positioning System(GPS)*, *Hill Climbing*, dan UML.

2.2.1 Distribusi Barang

Distribusi barang adalah rangkaian kegiatan penyaluran barang atau jasa dari produsen ke konsumen. dengan tujuan untuk memenuhi kebutuhan konsumen. Distribusi sering dikaitkan dengan masalah transportasi. Transportasi itu sendiri merupakan perpindahan manusia atau barang dari suatu tempat ke tempat lainnya dengan menggunakan sebuah kendaraan yang digerakan oleh manusia atau mesin, dimana hal ini disebabkan dalam melakukan proses distribusi sebuah perusahaan harus memikirkan bagaimana cara mengeluarkan biaya dan waktu yang paling minimal untuk proses penyaluran barang yang efektif [11]. Adapun model distribusi barang yang diterapkan pada PT. XYZ adalah sebagai berikut :

2.2.1.1 Model Distribusi (*Last-Mile Delivery*)

Last-Mile Delivery merupakan model jaringan distribusi dimana dalam pengiriman barangnya, pihak distributor atau *retailer* tidak menggunakan jasa pengiriman pihak ketiga, tetapi langsung dikirimkan oleh mereka dari rumah ke rumah [12].



Gambar2 3 Diagram alur *Last-Mile Delivery*

Ada dua faktor yang mempengaruhi kinerja pada model distribusi ini yaitu :

1. Faktor biaya
    a. Persediaan barang, model distribusi ini membutuhkan level persediaan barang yang lebih banyak dibanding dengan model distribusi lainnya. Dimana produk yang digunakan dalam kasus ini biasanya produk yang "bergerak cepat" karena dibutuhkan oleh pelanggan sesegera mungkin.
    b. Transportasi, diantara model distribusi yang ada biaya transportasi yang paling tinggi adalah model ini, terutama apabila terjadi pengiriman hanya kepada satu pelanggan dan jauh. Model distribusi ini akan dapat mengurangi biaya transportasinya secara signifikan apabila berada dikota-kota besar dengan padat penduduk, terutama apabila distributor mempunyai penjualan yang sangat besar dan mempunyai keberagaman produk yang besar.
    c. Fasilitas, dalam tahap ini barang biasanya dikirimkan dari pabrik ke gudang perusahaan dalam jumlah besar dan mengirimkan barang dari gudang ke pelanggan dalam jumlah yang sangat kecil. Apabila proses tersebut tidak ditangani dengan baik, maka akan berdampak negatif terhadap perusahaan.
    d. Informasi, model ini sangat membutuhkan infrastruktur informasi yang sangat baik antara distributor dan produsen (pabrik). Selain itu,

pelanggan juga sebaiknya dapat melihat kedalam proses pemesanan yang ada di pabrik, walaupun proses pemesanannya dilakukan melalui distributor. Dengan konsumen dapat mengetahui informasi pemesanan, selain konsumen merasa lebih aman, konsumen akan mendapatkan pengalaman yang baik [12].

2. Faktor layanan
    a. Waktu respon, model ini lebih cepat dibandingkan model distributor lainnya. Beberapa produk biasanya menjanjikan dikirimpada hari yang sama seperti waktu pemesanan.
    b. Keberagaman produk, biasanya keberagaman produknya lebih sedikit dibanding model lainnya.
    c. Ketersediaan produk, perusahaan akan lebih mudah dalam menjaga ketersediaan produk barangnya karena dalam memenuhi permintaan konsumen, perusahaan langsung melakukan permintaan barang ke pada produsennya sendiri.
    d. Pengalaman konsumen, dengan menggunakan sistem informasi yang tepat, maka model ini dapat memberikan pengalaman konsumen yang baik. Selain itu, produk yang dikirimkan akan terjamin kualitasnya, karena langsung dari produsen.
    e. *Time to market*, pada model ini, ketersediaan produk di pasar dapat dilakukan pada saat yang sama suatu produk dihasilkan oleh produsen.
    f. *Order visibility*, keterbukaan pemesanan (*order visibility*) tidak terlalu kompleks dibandingkan model lainnya mengingat proses pengirimannya biasanya dilakukan dalam hari yang sama. Informasi yang dibutuhkan adalah apakah barang sudah dikirim, barang ditunda atau dibatalkan.
    g. *Returnability*, pengembalian (*returnability*) model ini adalah yang terbaik,mengingat distributor dapat mengambil barang yang dikembalikan oleh konsumen ketika distributor mengirim barang lain ke konsumen [12].

2.2.2 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis Linux dan bersifat terbuka (*open source*). Android SDK (*Software Development Kit*) menyediakan *tools* dan API (*Application Programming Interface*) yang diperlukan bagi para pengembang untuk membuat dan mengembangkan aplikasi yang digunakan pada ponsel bersistem operasi Android dengan menggunakan bahasa pemrograman Java maupun Kotlin [13].

2.2.2.1 Sejarah Android

Perjalanan Android dimulai sejak Oktober 2003 ketika 4 orang pakar IT, Andi Rubin, Rich Minner, Nick Sears dan Chris White mendirikan Android.Inc, di California US. Visi Android untuk mewujudkan *mobile device* yang lebih peka dan mengerti pemiliknya, kemudian menarik raksasa dunia maya Google. Google kemudian mengakuisisi Android pada Agustus 2005. OS Android dibangun berbasis *platform* Linux yang bersifat *open source*. Dengan nama besar Google dan konsep *open souce* pada OS Android, tidak membutuhkan waktu lama bagi android untuk bersaing dan menyisihkan *Mobile OS* lainnya seperti *Symbian, Windows Mobile, Blackberry* dan iOS. [14].

2.2.2.2 Versi Android

Sistem operasi android terdiri dari beberapa versi dimana setiap versi android memiliki kode unik dimana huruf awalan versi diurutkan berdasarkan abjad, dan diambil dari nama makanan kecuali untuk version 1.0 dan 1.1 tidak dirilis dengan nama kode tersebut. Kode tersebut diterapkan mulai pada android 1.5 Cupcake 2009 [15]. Berikut adalah perkembangan versi android yang dapat dilihat pada tabel 2.1.

Tabel 2.1 Versi Android

Code Name	Version Number	Tanggal Rilis
Alpha	1.0	September 23, 2008
Beta	1.1	February 9,2009
Cupcake	1.5	April 27, 2009

Code Name	Version Number	Tanggal Rilis
Donut	1.6	September 15,2009
Eclair	2.0 - 2.1	October 26,2009
Froyo	1.1 2.2.3	May 20,2010
Gingerbread	2.3 – 2.3.7	December 6,2010
Honeycomb	3.0 3.2.6	February 22,2011
Ice Cream Sandwich	4.0 – 4.0.4	October 18,2011
Jelly Bean	4.1 – 4.3.1	July 9,2012
Kitkat	4.4 – 4.4.4	October 21,2013
Lollipop	5.0 – 5.1.1	November 12,2014
Marshmallow	6.0 – 6.0.1	October 5,2015
Nougat	7.0 – 7.1.2	August 22,2016
Oreo	8.0 – 8.1.0	Auguts 21,2017
Pie	9.0	Auguts 6,2018
Android Q	10.0	Auguts 22,2019

Dimana pada android 10 ini sudah tidak memakai nama makanan manis, kemungkinan era penamaan version android dengan tema makanan manis sudah berakhir. Dan pada android Q ini logo robot android sudah tidak memiliki badan, kaki, dan tangan melainkan hanya kepala robotnya saja.

2.2.3 *Weighted Product*

Metode *Weighted Product* (WP) adalah sebuah metode dari *Multiple Attribute Decision Making* (MADM). MADM adalah suatu metode yang digunakan untuk mencari alternatif optimal dari sejumlah alternatif dengan kriteria tertentu. Inti dari MADM adalah menentukan nilai bobot untuk setiap atribut, kemudian dilanjutkan dengan proses perangkingan yang akan menyeleksi alternatif yang sudah diberikan [16].

2.2.4 *Global Positioning System* (GPS)

Salah satu perlengkapan modern untuk navigasi adalah *Global Positioning System* (GPS) merupakan sistem navigasi berbasis satelit dimana perangkat dapat mengetahui posisi koordinat bumi secara tepat yang dapat secara langsung menerima sinyal dari satelit. Perangkat GPS modern menggunakan peta sehingga menjadi perangkat modern dalam navigasi di darat, perairan dan udara [17].

2.2.5 Metode *Hill Climbing Search*

Hill Climbing sering digunakan jika terdapat suatu fungsi *heuristic* yang baik untuk mengevaluasi *state*. Terdapat dua jenis *Hill Climbing* yang sedikit berbeda, yakni *Simple Hill Climbing* (SHC) dan *Steepest-ascent Hill Climbing* (SACH). *Simple Hill Climbing* (SHC), secara sederhana langsung memilih *new state* yang memiliki jalur yang lebih baik (curam) daripada jalur-jalur sebelumnya tanpa memperhitungkan jalur-jalur lain yang lebih "curam". Sedangkan *Steepest Ascent Hill Climbing* (SACH) sesuai dengan namanya, akan mengevaluasi semua *state* yang berada dibawah *current state* dan memilih *state* dengan jalur paling "curam" [18].

Secara garis besar, berikut langkah-langkah dari Metode *Hill Climbing* :

1. Dibentuk lintasan awal sebagai keadaan awal (*initial state*), kemudian dilakukan pengujian dari keadaan awal tersebut. Pengujian dilakukan dengan menghitung total jarak yang ditempuh pada initial state. Pengujian tersebut dilakukan untuk mendapatkan nilai heuristik sebagai pembanding terhadap nilai-nilai heuristik yang ada setelah dilakukan kombinasi pertukaran dua kota. Hasil dari pengujian tersebut adalah jarak minimal yang ditempuh pada permasalahan. Hasil tersebut akan digunakan sebagai pembanding hasil dari langkah berikutnya.
2. Setelah dibentuk lintasan awal sebagai keadaan awal, langkah selanjutnya yaitu melakukan kombinasi penukaran dua kota kemudian dilakukan pengujian seperti pada poin pertama, hingga mencapai kondisi *goal*, jika tidak maka pilihlah elemen yang lain hingga mencapai goal. Kondisi *goal* diartikan sebagai keadaan yang merupakan solusi optimum dari permasalahan dengan kondisi: panjang jalur baru < panjang jalur lama.

3. Jika kondisi *goal* telah tercapai maka kondisi tersebut adalah solusinya, jika tidak, maka bukan solusinya.

Algoritma untuk *Simple Hill Climbing* sebagai berikut :

1. Evaluasi *initial state*. Jika *initial state* adalah *goal state* maka jadikan *state* ini sebagai solusi dan keluar dari program. Jika bukan *goal state*, lanjutkan proses dengan *initial state* sebagai *current state*.
2. Ulangi sampai solusi ditemukan atau sampai tidak ada operator baru yang dapat diaplikasikan terhadap *current state*:
   a. Pilih operator yang belum diaplikasikan terhadap *current state* dan aplikasikan operator tersebut sehingga menghasilkan *new state*.
   b. Evaluasi *new state* :
      i. Jika *state* ini merupakan *goal state* maka jadikan *state* ini sebagai solusi dan keluar dari program. ii.
      ii. Jika *state* ini bukan *goal state* tetapi lebih baik dari *current state* maka jadikan state ini sebagai current state baru. iii.
      iii. Jika *state* ini tidak lebih baik dari *current state* maka kembali ke langkah 2.a [5].

Algoritma untuk Steepest Ascent Hill Climbing sebagai berikut :

1. Evaluasi *initial state*. Jika *initial state* adalah *goal state* maka jadikan *state* ini sebagai solusi dan keluar dari program. Jika bukan *goal state*, lanjutkan proses dengan *initial state* sebagai *current state*.
2. Ulangi sampai solusi ditemukan atau sampai tidak ada perubahan terhadap *current state*:
   a. Misalkan SUK adalah suatu state yang menjadi suksesor dari *current state.*
   b. Untuk setiap operator yang bisa dilakukan terhadap *current state*, kerjakan:
      i. Aplikasikan semua operator yang ada, untuk menemukan *new state.*
      ii. Evaluasi *new state*. Jika merupakan *goal state*, jadikan ini sebagai solusi dan keluar dari program. Jika bukan *goal state*, bandingkan dengan new state dengan SUK. Jika *new state* lebih baik dari SUK

maka ganti SUK dengan new state. Jika *new state* tidak lebih baik dari SUK, tidak perlu diganti.

3. Jika SUK lebih baik dari *current state* maka ganti *current state* dengan SUK [5].

2.2.6 UML

Pada perkembangan teknik pemrograman berorientasi objek, muncullah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek yaitu *Unified Modeling Language* (UML) [19].

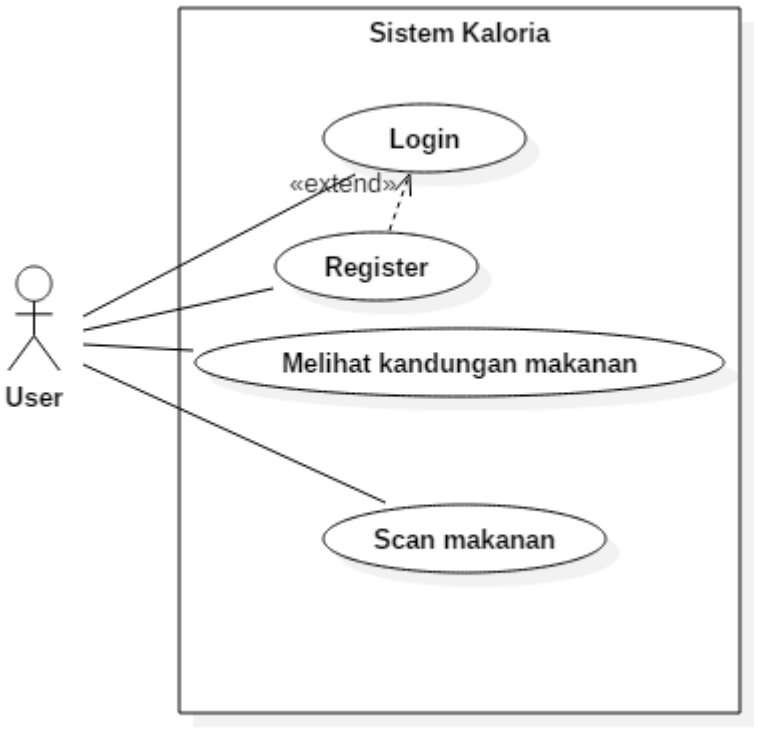
UML (*Unified Modelling Language*) merupakan bahasa pemodelan grafis yang digunakan untuk mendesain dan membantu pendeskripsian sistem perangkat lunak, khususnya sistem yang berorientasi objek. UML mencakup berbagai masalah yang meliputi spesifikasi, visualisasi, konstruksi dan dokumentasi berbagai jenis perangkat lunak sistem, oerangkat lunak non-sistem, dan model bisnis. Selain itu, dapat digunakan dalam pengembangan berbagai tahapan, mulai dari analisis kebutuhan sistem sampai implementasi sistem. UML memenuhi persyaratan objek analisis dan desain karena termasuk diagram alternatif untuk menjelaskan statis properti, penggunaan sistem atau komponen dan sistem arsitektur [20].

UML bisa juga berfungsi sebagai sebuah *blue print* atau cetak biru karena sangat lengkap dan detail. Dengan cetak biru ini maka akan bisa diketahui informasi secara detail tentang koding program atau bahkan membaca program dan menginterpretasikan kembali ke dalam bentuk diagram (*reserve engineering*) [21].

2.2.6.1 *Use Case Diagram*

Use case diagram adalah diagram yang digunakan untuk menggambarkan fungsional yang diharapkan dari sebuah sistem. Dimana diagram tersebut menggambarkan secara ringkas siapa yang menggunakan sistem dan apa saja yang bisa dilakukannya digambarkan melalui hubungan antara aktor, *use case*, dan *system*. *Diagram use case* berkonsentrasi pada menunjukkan tujuan sistem (*use case*) dan pengguna (aktor) biasanya juga disebut diagram kasus penggunaan dimana diagram tersebut menunjukkan sistem dalam lingkungannya (konteks) dari

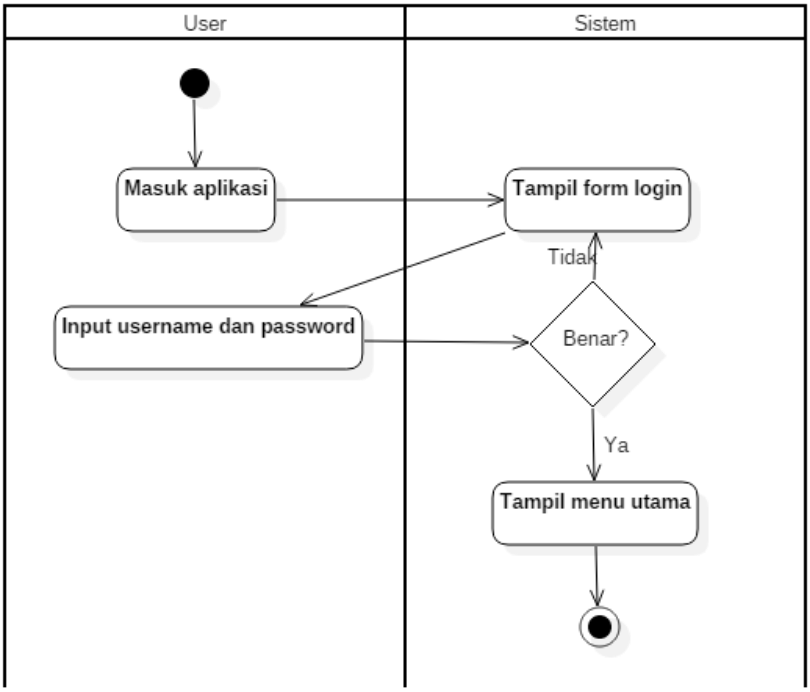
sistem dan aktor di sekitarnya [22]. Salah satu contohnya dapat dilihat pada gambar 2.3.



Gambar 2.4 *Use Case Diagram*

2.2.6.2 *Activity Diagram*

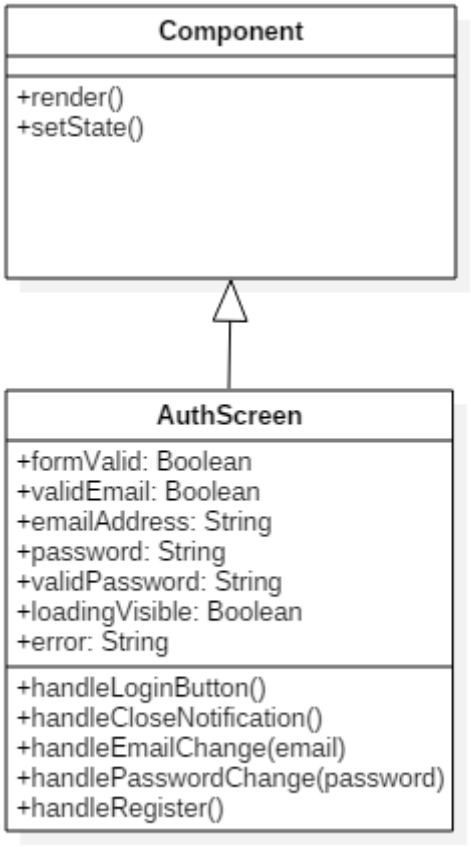
Activity diagram adalah diagram yang menggambarkan *workflow* (alur kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Diagram aktivitas sangat berguna ketika ingin menggambarkan pekerjaan yang mengalir melalui proses bisnis [22] . Salah satu contohnya dapat dilihat pada gambar 2.4



Gambar 2.5 *Activity Diagram*

2.2.6.3 *Class Diagram*

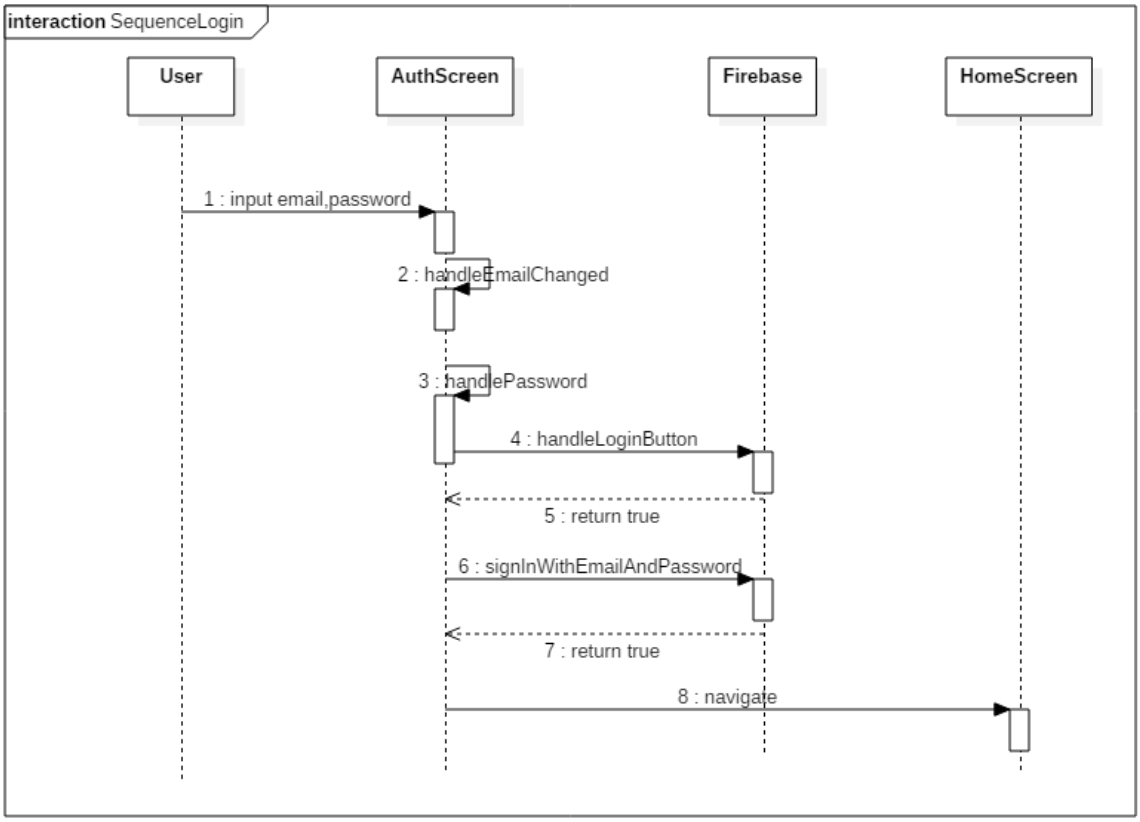
Class diagram adalah diagram yang menggambarkan struktur sistem dari segi pendefinisian kelas -kelas yang akan dibuat untuk membangun sistem. Dimana kelas itu sendiri memiliki 3 bagian utama yaitu *attribute*, *operation*, dan *name*. kelas kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. [22]. Salah satu contohnya dapat dilihat pada gambar 2.5.



Gambar 2.6 *Class Diagram*

2.2.6.4 *Sequence Diagram*

Sequence diagram adalah diagram yang menggambarkan tingkah laku objek dan mengetahui komunikasi/pesan diantara objek objek tersebut. *Sequence diagram* dapat membantu menggambarkan setiap peristiwa secara berurutan dalam sebuah sistem atau perangkat lunak. [22]. Salah satu contohnya dapat dilihat pada gambar 2.6.



Gambar 2.7 *Sequence Diagram*

2.3 Perangkat Lunak Pendukung

Berikut adalah perangkat lunak pendukung dalam penunjang pembangunan aplikasi yang akan dibangun.

2.3.1 Android Studio

Android Studio adalah IDE (*Integrated Development Environment*) resmi untuk pengembangan aplikasi Android dan bersifat *open source* atau gratis. Peluncuran Android Studio ini diumumkan oleh Google pada 16 Mei 2013 pada *event* Google I/O *Conference* untuk tahun 2013. Sejak saat itu Android Studio menggantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi android.

Android studio sendiri dikembangkan berdasarkan Intellij IDEA yang mirip dengan Eclipse disertai dengan ADT *plugin* (*Android Development Tools*). Android studio memiliki fitur [23]:

a. Projek berbasis pada *Gradle Build*

b. *Refactory* dan pembenahan bug yang cepat
c. *Tools* baru yang bernama “Lint” diklaim dapat memonitor kecepatan, kegunaan, serta kompetibelitas aplikasi dengan cepat
d. Mendukung *Proguard* dan *App-signing* untuk keamanan
e. Memiliki GUI aplikasi android lebih mudah
f. Didukung oleh Google Cloud Platform untuk setiap aplikasi yang dikembangkan.

2.3.2 *Visual Studio Code*

Visual Studio Code adalah editor kode sumber yang dikembangkan oleh Microsoft untuk windows, linux maupun macOS. Dimana mencangkup dukungan untuk :

a. *Cross Platform,* dimana dapat bekerja untuk berbagai platform seperti windows, linux dan macOS tanpa khawatir belajar *coding tools* yang sama untuk sistem yang berbeda-beda.
b. *Light Weight,* dimana anda dapat mengontrol sepenuhnya bahasa, tema, *debugger* , *commands* dan lain lainnya sesuai kebutuhan. Ini dapat dilakukan melalui *extentions* untuk bahasa yang sedang banyak digunakan seperti *python, node.js, java* dan lainnya.
c. *Powerful Editor,* dengan adanya dukungan ini maka anda bias lebih efektif seperti membuat kode *snippets, IntelliSense, auto correct* dan *formatting*.
d. *Code Debugging,* dukungan ini membantu dalam hal melakukan debugging pada kode dengan cara mengawasi kode, *variable, call stack* dan *expression*.
e. *Source Control,* sudah terintegrasi dengan Git dan penyedia *source code* control lainnya.
f. *Integrated Terminal,* istilah *Multiple Windows* dan ALT+TABS sudah tidak diperlukan lagi, karena dengan dukungan ini dapat melakukan *command-line task* dengan cepat dan dapat membuat banyak terminal di dalam editor.

2.3.3 Postman

Postman adalah platform kolaborasi untuk mengembangkan sebuah *Application Programming Interface* (API). Dimana *postman* ini menyederhanakan setiap langkah pembangunan API dan mempermudah kolaborasi sehingga dapat

membuat API yang lebih efektif dan efisien. Berikut fitur yang disediakan oleh postman :

1. *Sharing Collection API for Documentation*
2. *Testing API*
3. *Realtime Collaboration Team*
4. *Monitoring API*
5. *Integration*

2.3.4 Java

Bahasa pemrograman Java merupakan salah satu dari sekian banyak bahasa pemrograman yang dapat dijalankan di berbagai sistem operasi, termasuk telepon genggam. Bahasa pemrograman ini pertama kali dibuat oleh *James Gosling* saat masih bergabung *Sun Microsystem.* Bahasa pemrograman ini merupakan pengembangan C++, saat ini Java merupakan bahasa pemrograman yang paling populer digunakan dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

Kelebihan Java dari bahasa pemrograman yang lain adalah bisa dijalankan di berbagai jenis sistem operasi sehingga dikenal juga bahasa pemrograman *multiplatform,* bersifat pemrograman berorientasi *object* (PBO), memiliki *library* yang lengkap [24].

2.3.5 PHP (*Hypertext Prepocessor*)

Hypertext Preprocessor atau PHP merupakan bahasa pemrograman berbasis web yang memiliki kemampuan untuk memproses data dinamis. PHP dikatakan sebagai sebuah *server-side embedded script language* artinya sintaks-sintaks dan perintah yang diberikan akan sepenuhnya dijalankan oleh server tetapi disertakan pada halaman HTML biasa [13].

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (Situs Personal). PHP dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP bernama *Form Interpreted* (F1) berupa sekumpulan script yang digunakan untuk mengolah data formulir dari web.

Berikut ini beberapa kelebihan dari bahasa pemrograman PHP :

1. Mudah untuk dikonfigurasi.
2. Relatif mudah dipahami , karena terdapat banyak referensi untuk dipelajari.
3. Bahasa pemrograman PHP dapat disisipkan kedalam HTML.
4. PHP bersifat open source alias gratis yang dimana dapat digunakan diberbagai system operasi.
5. Masih dikembangkan sampai saat ini.

2.3.6 JSON

JSON (dibaca: “Jason”), singkatan dari *JavaScript Object Notation* adalah suatu format ringkas pertukaran data komputer. Formatnya berbasis teks dan terbaca manusia serta digunakan untuk merepresentasikan struktur data sederhana dan larik asosiatif (disebut objek). Format JSON sering digunakan untuk mentransmisikan data terstruktur melalui suatu koneksi jaringan pada suatu proses yang disebut *serialisasi* [13].

2.3.7 *Web Service*

W3C mendefinisikan *web service* sebagai sebuah sistem perangkat lunak yang dirancang untuk mendukung komunikasi dan interaksi antar mesin ke mesin (*Machine to Machine*) melalui sebuah *network* (jaringan). *Web service* juga termasuk *WEBAPIs* yang dapat diakses melalui jaringan seperti misalnya internet, dan dieksekusi melalui sebuah sitem jarak jauh sesuai dengan layanan yang diminta [25].

2.3.8 API (*Application Programming Interface*)

API merupakan *software interface* yang terdiri atas kumpulan instruksi yang disimpan dalam bentuk *library* dan menjelaskan bagaimana agar suatu *software* dapat berinteraksi dengan *software* lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari analogi tersebut, rumah merupakan *software* yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari *software* tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut [25].

2.3.9 MySQL

MySQL didefinisikan nama database server. Database server adalah server yang berfungsi untuk menangani database. Database adalah suatu pengorganisasian data. Dengan menggunakan MySQL, pemrogram dapat menyimpan data kemudian data bisa diakses dengan cara yang mudah dan cepat [19].

MySQL menggunakan bahasa dasar yaitu SQL (*Structured Query Language*) dimana terdiri dari 3 bagian yaitu DDL(*Data Defination Language*), DML(*Data Manipulation Language*), dan DCL(*Data Control Language*).

2.3.9.1 *Data Defination Language* (DDL)

Tujuan data *definition language* adalah untuk membuat struktur sebuah database, dimana ada lima jenis perintah dasar seperti :

1. Perintah *CREATE*

   Sebuah perintah yang digunakan untuk membuat sebuah database baru, baik itu berupa table atau sebuah kolom baru.
2. Perintah *ALTER*

   Sebuah perintah yang digunakan untuk mengubah struktur *table* yang sebelumnya sudah ada, Bisa jadi nama *table*, penambahan kolom, mengubah, maupun menghapus kolom serta menambahkan atribut lainnya.
3. Perintah *RENAME*

   Sebuah perintah yang digunakan untuk mengubah sebuah nama di *table* ataupun kolom yang ada.
4. Perintah *DROP*

   Sebuah perintah yang digunakan untuk menghapus baik itu berupa database,*table* maupun kolom hingga index.
5. Perintah *SHOW*

   Sebuah perintah yang digunakan untuk menampilkan sebuah stuktur *table*.

2.3.9.2 *Data Manipulation Language* (DML)

Data Manipulation language ini merupakan sebuah perintah yang bertujuan untuk memanipulasi data yang ada dalam sebuah database. Perintah DML juga terbagi ke dalam empat jenis yaitu :

1. Perintah *INSERT*

Perintah ini digunakan untuk memasukan sebuah *record* atau data baru ke dalam sebuah *table* database.

2. Perintah *SELECT*

   Perintah ini digunakan untuk menampilkan maupun mengambil sebuah data pada *table*. Data yang diambil juga tidak hanya terbatas pada satu *table* bahkan bisa lebih dari satu.

3. Perintah *UPDATE*

   Perintah ini digunakan untuk ingin melakukan pembaruan data disebuah *table*.

4. Perintah *DELETE*

   Perintah ini digunakan untuk menghapis sebuah *record* dalam sebuah *table*.

2.3.9.3 *Data Control Language*(DCL)

Data Control Language merupakan perintah SQL yang digunakan khususnya untuk mengatur hak apa saja yang dimiliki oleh pengguna. Baik itu hak terhadap sebuah database ataupun pada *table* maupun *field* yang ada. Perintah DCL ini terbagi dua yaitu :

1. Perintah *GRANT*

   Perintah ini biasanya digunakan ketika admin database ingin memberikan hak akses ke *user* lainnya, dalam hal ini admin dapat memberikan akses mengenai perintah dalam DML diatas.

2. Perintah *REVOKE*

   Perintah ini biasanya digunakan untuk mencabut mapaun menghapus hak askes seseorang yang awalnya diberikan akses oleh admin database melalui perintah *GRANT* sebelumnya.

2.3.10 MapBox

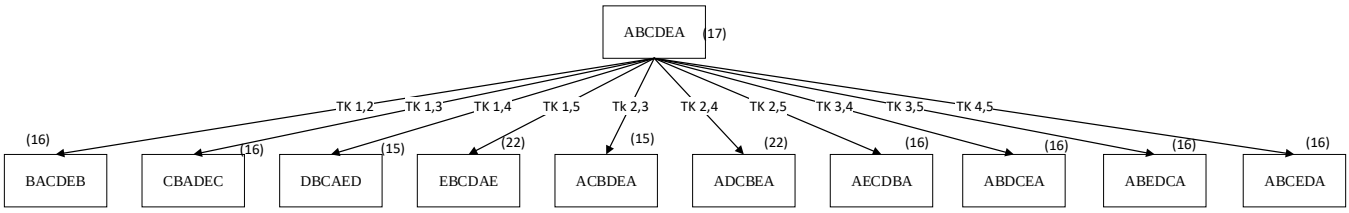
MapBox merupakan *platform* penyedia layanan peta untuk *mobile* aplikasi ataupun website, Mapbox didirikan oleh sebuah tim kecil di Washington DC pada tahun 2010. Semenjak didirikan, Mapbox telah banyak digunakan oleh perusahaan contohnya adalah Foursquare pada tahun 2012, dan USA Today ketika dilaksanakannya pemilihan umum pada tahun 2012 [26].

Pada tahun 2014 Mapbox merilis Android SDK yang dapat diimplementasikan pada perangkat Android [26]. Mapbox Android SDK menyediakan beberapa fitur yang dapat digunakan pada aplikasi Android yang ingin diciptakan, yaitu :

1. Geocoding API & *autocompletion*
2. *Direction* API & *route display*
3. *Navigation* SDK
4. *Static Maps* API & *Android integration*
5. *Geospatial analysis functionality*, diadaptasi dari projek Turf
6. *Map Matching* API
7. *Line Simplification*

2.4 Keterkaitan Antara Metode Yang Digunakan Dengan Optimalisasi Distribusi Barang

Dalam proses pemecahan masalah di PT. XYZ ini menggunakan metode *Hill Climbing* dimana lebih tepatnya menggunakan metode *Steepest Ascent Hill Climbing* metode ini membantu untuk memberikan daftar pengiriman barang berdasarkan keterdekatan jarak dari perhitungan garis lintang(*latitude,longitude*) setiap lokasi pengiriman kepada konsumen, cara kerja metode ini sangatlah tepat untuk menentukan jalur pengiriman yang efektif karena pada metode ini setiap lokasi pengiriman akan dihitung keterdekatannya dengan lokasi pengiriman lainnya dimana pada akhirnya metode ini akan menghasilkan beberapa jalur dengan total jarak yang berfariasi, kita hanya tinggal memilih jalur terpendek dimana itu adalah jalur teroptimal yang dihitung dengan metode ini.



Gambar 2 8 Diagram alur *Steepest Ascent Hill Climbing*

Dapat dilihat dari contoh gambar diatas bahwa pencarian jalur optimal berdasarkan penukaran dua buah kota. Dan perhitungannya sangat efektif karena secara logis jika kita memiliki titik awal A dan akan mengirim ke titik B,C,D,E akan dibuat jalur maka terbentuk jalur ABCDE dalam perhitungan peluang kemungkinannya berarti 5x5 ada dua puluh lima kemungkinan perhitungan (25). Dapat dilihat pada tabel 2.2

Tabel 2.2 Peluang kemungkinan

	A	B	C	D	E
A	(A,A)	(A,B)	(A,C)	(A,D)	(A,E)
B	(B,A)	(B,B)	(B,C)	(B,D)	(B,E)
C	(C,A)	(C,B)	(C,C)	(C,D)	(C,E)
D	(D,A)	(D,B)	(D,C)	(D,D)	(D,E)
E	(E,A)	(E,B)	(E,C)	(E,D)	(E,E)

Tetapi dari metode tersebut hanya menghitung 10 kemungkinan. Hal itu dikarenakan dalam metode ini menggunakan perhitungan kombinasi dimana rumusnya adalah $\frac{n!}{2!(n-2)!}$. Sehingga jika kita memiliki jalur pengirim dari titik kantor ke titik konsumen ABCDE berarti kita mempunyai 5(n) tujuan pengiriman maka mendapatkan kombinasi sebanyak $\frac{5!}{2!(5-2)!}$ =10 . Sepuluh kombinasi ini akan dipakai sebagai operator penukar dua buah kota yaitu :

1. Tukar sekolah 1,2 (menukar urutan posisi tujuan ke-1 dengan tujuan ke-2).
2. Tukar sekolah 1,3 (menukar urutan posisi tujuan ke-1 dengan tujuan ke-3).
3. Tukar sekolah 1,4 (menukar urutan posisi tujuan ke-1 dengan tujuan ke-4).
4. Tukar sekolah 1,5 (menukar urutan posisi tujuan ke-1 dengan tujuan ke-5).
5. Tukar sekolah 2,3 (menukar urutan posisi tujuan ke-2 dengan tujuan ke-3).
6. Tukar sekolah 2,4 (menukar urutan posisi tujuan ke-2 dengan tujuan ke-4).
7. Tukar sekolah 2,5 (menukar urutan posisi tujuan ke-2 dengan tujuan ke-5).
8. Tukar sekolah 3,4 (menukar urutan posisi tujuan ke-3 dengan tujuan ke-4).
9. Tukar sekolah 3,5 (menukar urutan posisi tujuan ke-3 dengan tujuan ke-5).
10. Tukar sekolah 4,5 (menukar urutan posisi tujuan ke-4 dengan tujuan ke-5).

Terlihat pada gambar diatas ACBDEA adalah jalur optimal pada level pertama jika ingin mengoptimalkannya jalurnya lagi ulangi hingga tidak ada jalur optimal lainnya. Tetapi sebenarnya pada tahap ini saja sudah optimal. Karena jika lanjut ke level 2 sudah tidak terurut lokasi awal dan lokasi tujuannya. Dilevel pertama juga kita harus memfilter jalur dimana lokasi harus mulai dari titik kantor pengiriman bukan dari titik konsumen. Pada akhirnya jalur optimal ACBDEA ini kita gunakan untuk parameter MapBox, yang nantinya MapBox akan memberikan arah jalur berdasarkan hasil pengurutan titik kordinat menggunakan *Steepest Ascent Hill Climbing* yaitu jalur kordinat ACBDEA.