BAB II
LANDASAN TEORI

2.1 Tinjauan Studi

Dalam penelitian ini menggali informasi dan penelitian sebagai bahan referensi, baik mengenai permasalahan, metode, dan hasil yang fungsinya untuk memudahkan peneliti dalam melakukan penelitiannya sesuai dengan tema dan membuat sistem yang baru dan bermanfaat. Pada tiga jurnal yang peneliti gunakan sebagai bahan referensi untuk acuan mengambil metode yang akan digunakan.

Pada jurnal pertama oleh Dwi Purnomo (2017) yang berjudul *"Model Prototyping pada Pengembangan Sistem Informasi"*. Banyak yang tidak tahu dan kurang pahamnya pengguna terhadap proses pengembangan sintem yang merupakan tugas seorang pengembang dalam menjembatani maksut yang dikehendaki secara bahasa umum untuk di tindak lanjut ke bahasa teknis sistem, walau pengembang terkadang memperhatikan algoritma alur sistem, perangkat bantu sistem, ataupun modal *interface* yang akan diwujudkan. Jadi untuk mengatasi masalah ketidaktahuan dari pengguna atau pemilik dengan pengembang, maka harus ada kerjasama yang baik antara kedua pihak sehingga pengembang bisa mengetahui dengan baik apa yang diharapkan dari pihak pengguna dengan tidak meninggalkan landasan teknik pengembangan, sehingga pengguna akan memenuhi kebutuhan sistem yang akan dikembangkan. Pada akhirnya akan dihasilkan sebuah sistem yang sesuai dengan kebutuhan pengguna serta pemahaman yang tepat dan sesuai dengan penjadwalan pengerjaan sistem yang sudah disepakati.[4]

Pada jurnal kedua oleh Christian R.Z Mamuaja, Don R.G Kabo, Nadya Kamasi (2018), yang berjudul "*Rancang Bangun Aplikasi Pemesanan Bahan Bangunan Berbasis Android pada Toko Walian Jaya Kota Tomohon*". Terkadang membuat penjual dan calon pembeli bingung untuk menentukan tempat maupun bahan yang akan dibeli atau dijual. Permasalahan terbesar yang dihadapi adalah kurangnya sarana kendaraan, kemungkinan calon konsumen tidak mau menunggu lama. Metode yang digunakan adalah *Prototyping* sebagai metode merancang aplikasi. Dengan hasil aplikasi pemesanan bahan bangunan yang dilengkapi dengan kemampuan baru karena akan di padukan dengan teknologi untuk pengumpulan data dan pengolahan data, dimana data yang sebelumnya dikumpulkan

menggunakan kertas dan sekarang menggunakan media digital yaitu dengan menggunakan sistem pemesanan secara *online* dan menggunakan sistem *Pre-Order*.[5]

Pada Jurnal ketiga oleh Arif Setiawan, Toufan Diansyah Tambunan, S.T., M.T., Robbi Hendriyanto, S.T., M.T. (2016), yang berjudul "*Android Augmented Reality Untuk Menampilkan Katalog Furniture Secara Tiga Dimensi (3D) Berdasarkan Objek Marker*", penelitian menjelaskan bahwa citra dari gambar *furniture* dari sebuah katalog dengan melihat representasi tiga dimensi dapat ditampilkan dari segala arah dan menampilkan informasi detail dari produk. Aplikasi tersebut diharap dapat memudahkan pengguna untuk mencari produk serta bentuk spesifik dari *furniture* yang di cari tanpa harus bersusah payah memindahkan benda aslinya karena cukup hanya dengan benda virtual yang mewakili wujud aslinya. Peneliti menggunakan metode *Luther* (1994) sebagai metode pengembangan aplikasi, pada perancangan aplikasi menggunakan *Use Case, Class Diagram, Sequnce Diagram,* dan *Activity Diagram.* Pada tahap akhir peneliti menggunakan Tahap *User Acceptance Test* (UAT) dan mencoba aplikasi pada sepuluh orang yang berkecimpung di bidang *furniture* diharapkan mendapatkan penilaian baik terhadap aplikasi yang sedang dibuat.[6]

2.2 Tinjauan Pustaka

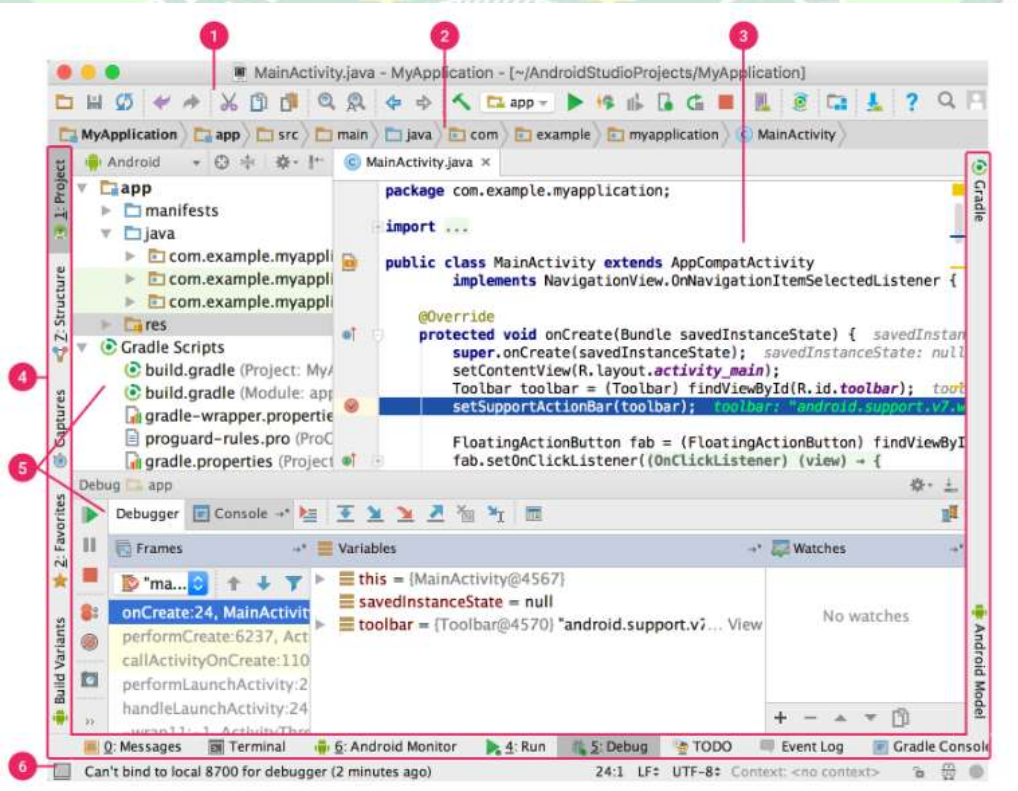
2.2.1 Android Studio

Android Studio merupakan IDE (*Integrated Development Environment*) resmi untuk untuk *platform Android*. *Android Studio* ini di umumkan pada tanggal 16 Mei 2013 pada konferensi *Google I/O* oleh Produk Manager *Google, Elite Power*. *Android Studio* bersifat *free* dibawah *Apache Licence 2.0*. *Android Studio* diawali dengan versi 0.1 pada bulan Mei 2013, kemudian dibautkan versi 0.8 pada bulan Juni 2014. Yang paling stabil dirilis pada bulan Desember 2014, lalu dimulai dari versi 1.0 Berbasiskan *JetBrain's Intellij IDE* dan di desain khusus untuk *Android Development*.[7]

Android Studio sendiri dikembangkan berdasarkan *IntelliJ IDEA* yang mirip *Eclipse* disertai dengan *ADT plugin* (*Android Development Tools*). *Android Studio* memiliki fitur sebagai berikut :

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 And App-signing* untuk keamanan.
e) Memiliki GUI aplikasi android lebih mudah.
f) Didukung oleh *Google Cloud Platform* untuk setiap aplikasi yang dikembangkan.

Namun dari segi grafik *Android Studio* lebih unggul dibandingkan *Eclipse*, tetapi *Android Studio* jauh lebih berat dibandingkan dengan *Eclipse* karena membutuhkan spesifikasi komputer yang cukup tinggi, dan *Android Studio* dikhususkan untuk pengembangan *Android Studio* saja dibandingkan *Eclipse* (tidak multi programming), jadi *Android Studio* lebih nyaman dalam pengkodingan untuk membuat atau mengembangkan sebuah aplikasi.



Gambar 2.1 Jendela utama *Android Studio*

Jendela utama pada *Android Studio* terdiri dari beberapa bidang logika yang diidentifikasi pada gambar :

1. **Bilah Alat** : memungkinkan *developer* untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan program Android.
2. **Bilah Navigasi** : membantu *developer* bernavigasi antara proyek dan membuka file untuk di edit. Bilah ini memberikan tampilan struktur yang ringkas dalam jendela *Project*.
3. **Jendela Editor** : tempat *developer* membuat dan memodifikasi kode bergantung pada jenis file, editor dapat berubah.. Seperti saat melihat file tata letak, editor menampilkan *Layout Editor*.
4. **Bilah Jendela Alat** : muncul saat di luar jendela IDE dan berisi tombol yang memungkinkan *developer* meluaskan atau mengecilkan jendela alat.
5. **Jendela Alat** : memberi *developer* akses ke tugas tertentu seperti pengelolaan proyek, pencarian atau penelusuran, kontrol versi, dan sebagainya.
6. **Bilah Status** : menampilkan status proyek dan IDE itu sendiri serta setiap peringatan.

2.2.2 Android

Android merupakan sistem operasi berbasis *Linux* untuk telepon pintar terpopuler yang dikeluarkan oleh *Google*. *Android* juga memiliki fitur dan tampilan baik serta digunakan alat multimedia dan dapat mengoperasikan perangkat keras seperti sersor dan lain sebagainya. Sejak rilis tahun 2008, hingga 2018 *Android* sudah merilis beberapa versi dengan versi-versi terbaru yaitu *Android Pie*. Pada tahun 2016 *Android* sudah memiliki lebih dari 2 miliar pengguna aktif tiap bulannya. Lisensi kode sumber *Android* memiliki lisensi *open source*. Sehingga, dapat menarik para komunitas pengembangan untuk menggunakan *Android* dalam projek komunitasnya.[7]

Tabel 2.1 Versi Android

No	Nama Kode	No. Versi	Tanggal Rilis	API Level
1	Alpha	-	Awal Tahun 2007	-
2	Beta	-	November 2007	-

3	Angel cake	1.0	23 September 2008	1
4	Bettenberg	1.1	9 Februari 2009	2
5	Cupcake	1.5	27 April 2009	3
6	Donut	1.6	15 September 2009	4
7	Éclair	2.0 – 2.1	26 Oktober 2009	5-7
8	Froyo	2.2 – 2.3	20 Mei 2010	8
9	Gingerbread	2.3 – 2.3.7	6 Desember 2010	9-10
10	Honeycomb	3.0 – 3.2.6	22 Februari 2011	11-13
11	Ice Cream Sandwich	4.0 – 4.0.1	18 Oktober 2011	14-15
12	Jelly Bean	4.1 – 4.3.1	9 Juli 2012	16-18
13	Kitkat	4.4 – 4.4.4	31 Oktober 2013	19-20
14	Lollipop	5.0	15 Oktober 2014	21-22
15	Marshmallow	6.0	5 Oktober 2015	23
16	Nougat	7.0	22 Agustus 2016	24

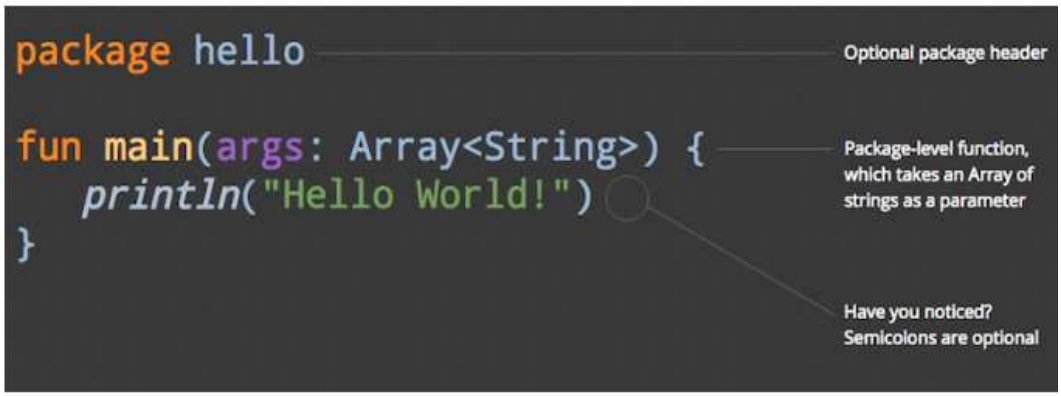
2.2.3 Kotlin

Kotlin merupakan Bahasa pemrograman dengan pengetikan statis dan dapat berjalan pada mesin *Virtual Java,* atau menggunakan *Compiler LLVM* yang dapat menyusun ke dalam bentuk kode sumber *JavaScript*. Memiliki keunggulan untuk mudah dipelajari, sederhana, dan efisien. *Kotlin* diumumkan sebagai salah satu Bahasa pemrograman yang dapat digunakan untuk membuat aplikasi *Android* pada *Google* I*/O* 2017 berdampingan dengan *Java* dan *C++* yang telah terlebih dahulu sebagai Bahasa resmi untuk pengembangan aplikasi *Android*.[8]

Berikut adalah beberapa perbaikan dari *Java* yang ada di *Kotlin* :

- *Null reference* kini dikontrol oleh *type sistem*
- Tidak ada *raw type*
- *Array* di *Kotlin* tidak berubah
- *Kotlin* memiliki *function type* yang layak
- *Use-site variance* tanpa perlu *wildcard*

- *Kotlin* tidak memiliki *checked exeption*



Gambar 2.2 Contoh koding *Kotlin* lebih sederhana dari *Java*

2.2.4 Firebase

Firebase merupakan BaaS (*Backend as a Service*) suatu layanan dari google untuk memudahkan para pengembang aplikasi untuk mengembangkan aplikasinya. Dengan adanya *firebase app, developer* bisa fokus mengembangkan aplikasi tanpa harus memberikan *effort* yang besar untuk urusan *backend*.

Pada tahun 2014, Google mengaku sisi perusahaan yang berbasis di San Fransisco bernama *Firebase, Inc*. Dengan menyediakan berbagai solusi pengembangan yang dirancang untuk integrase fitur berbasis *cloud* ke dalam aplikasi sululer dan web. Setelah memberi perusahaan, *Google* menggabungkan layanan yang disediakan oleh *Firebase* dengan sejumlah fitur pelengkap yang sebelumnya termasuk dalam bagian dalam dari *Google Cloud Platform*. Fitur yang sebelumnya tergabung dari dua *platform* adalah apa yang sekarang dikenal sebagai *Firebase.*[9]

Adapun fitur yang dimiliki firebase, diantaranya :

1. *Google Analityc*

   Anilityc ini memberikan data seputar perilaku pengguna pada aplikasi *Android* dan *iOS* agar pengguna dapat mengambil keputusan yang lebih baik tentang informasi produk dan pengoptimaalan pemasaran.

2. *Real-Time Database*

   Untuk menyimpan dan sinkronkan data antara pengguna dan perangkat secara *realtime* menggunakan *database noSQL* yang di *hosting* secara *cloud*.

3. *Authentication*

   Untuk mengelola pengguna dengan cara yang mudah dan aman.

4. *Cloud Storage*

   Untuk menyimpan dan membagikan gambar, audio, video, atau konten lain secara mudah dengan menyimpan objek yang handal, sederhana, dan hemat biaya.

5. *Hosting*

   Memudahkan *hosting Web* statis dengan fitur yang dibuat khusus untuk *Web* aplikasi modern.

Ada pula struktur *Firebase* yang berbeda dengan *MySQL* diantaranya :

1. *Firebase Realtime Database* bertipe *NoSQL*, sedangkan *MySQL* bertipe RDBMS.


```
{
  "catatanBerat": {
    "1354554000,ffNCPzXkO1XJv2JenczhHmmAZ9jZ": {
      "fbId": "ffNCPzXkO1XJv2JenczhHmmAZ9jZ",
      "weight": 85
    },
    "1354554000,ffNCPzXkO1XJv2JenczhHmlAZ9jY": {
      "fbId": "ffNCPzXkO1XJv2JenczhHmlAZ9jY",
      "weight": 85
    }
  },
  "profilFB": {
    "SFFX82dLeqQegEa0U9nYG3mwf2u4": {
      "email": "nyotpia@gmail.com",
      "fbId": "SFFX82dLeqQegEa0U9nYG3mwf2u4",
      "name": "Donita Buana",
      "profPicUrl": "https://graph.facebook.com/402677183492676/picture?height=500",
      "score": 150000,
      "ts": {
        "timestamp": 1514985701130
      }
    },
    "SFYw0TlzbSRhWOTXdbrVsQdFFjj3": {
      "email": "",
      "fbId": "SFYw0TlzbSRhWOTXdbrVsQdFFjj3",
      "name": "Risma Hutabarat",
      "profPicUrl": "https://graph.facebook.com/1924094845287088/picture?height=500",
      "score": 220000,
      "ts": {
        "timestamp": 1515413357505
      }
    }
  }
}
```


Gambar 2.3 Contoh koding dari *Firebase* dengan format yang sama yaitu *JSON*

2. *Firebase Realtime Database* bersifat *Realtime*, sedangkan *MySQL* tidak *Realtime*.

3. Dengan *Firebase Realtime Database* tidak perlu membuat koding di sisi server.
4. Koding yang dipakai *Firebase* untuk mengolah data kompleks yang lebih banyak dan rumit daripada koding yang dipakai untuk mengolah data *MySQL*.
5. *Firebase Realtime Database* punya versi berbayar, sedangkan *MySQL* tidak berbayar atau gratis sepenuhnya.

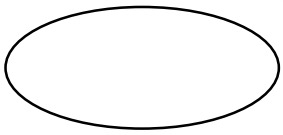
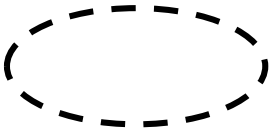
2.2.5 Unified Modeling Language

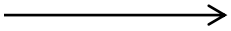
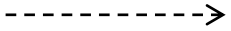
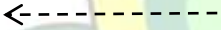
UML (*Unified Modelling Language*) merupakan visualisasi dan dokumentasi hasil analisa dan desain sintak dalam memodelkan sistem secara visual, serta merupakan alat perancangan sistem yang berorientasi pada objek. UML (*Unified Modeling Language*) ditujukan untuk membantu tim pengembang berkomunikasi, eksplorasi potensi desain, dan validasi desain arsitektur perangkat lunak. UML (*Unified Modeling Language*) memiliki tiga kategori utama yaitu *behaviour* diagram, struktur diagram, dan *interaction* diagram, yang dimana pada setiap kategori diagram menjelaskan arsitektur sistem dan saling terintegrasi.[10]

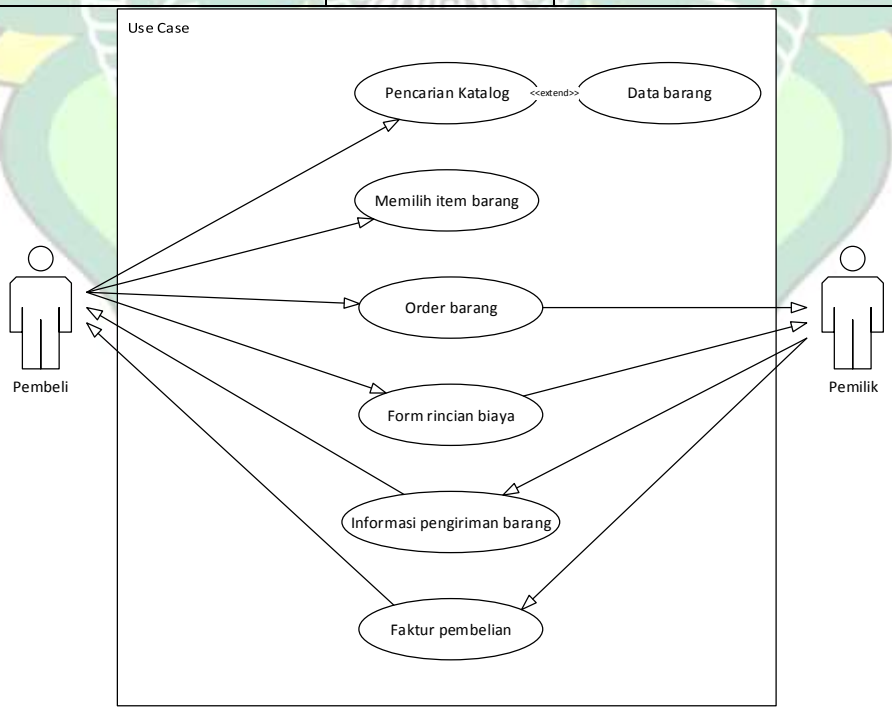
2.2.6 Use Case Diagram

Use Case Diagram merupakan deskripsi fungsi dari sistem berdasarkan perspektif pengguna, dan merupakan sebuah dokumentasi yang digunakan sebagai proses analisa untuk menangkap permintaan sistem dan memahami sistem bekerja. *Use Case Diagram* menetapkan apa yang dikerjakan oleh sistem seperti kebutuhan fungsional dan tidak untuk menentukan kebutuhan non-fungsional.[10] Berikut adalah bentuk simbol dan fungsi dari sistem *use case* tersebut :

Tabel 2.2 Simbol *Use Case Diagram*

No	Simbol	Nama	Keterangan
1.		Use Case	Deskripsi dari urutan aksi yang ditampilkan sistem.
2.		Collaboration	Interaksi elemen lain yang bekerja sama untuk

			menyediakan perilaku yang besar dari jumlah yang ada.
3.		Generalization	Hubungan pada objek anak berbagi perilaku dan struktur data dari objek induk.
4.		Include	Memungkinkan 1 use case menggunakan fungsionalitas yang di sediakan oleh use case yang lain.
5.		Extend	Memungkinkan 1 use case secara optimal menggunakan fungsilitas dari use case yang lain.
6.		Association	Menghubungkan antara objek satu ke objek yang lain.
7.		Aktor	Pengguna sistem yang berhubungan dengan sistem lain atau waktu



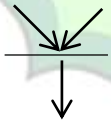
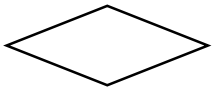
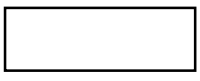
Gambar 2.4 Contoh *Use Case Diagram*

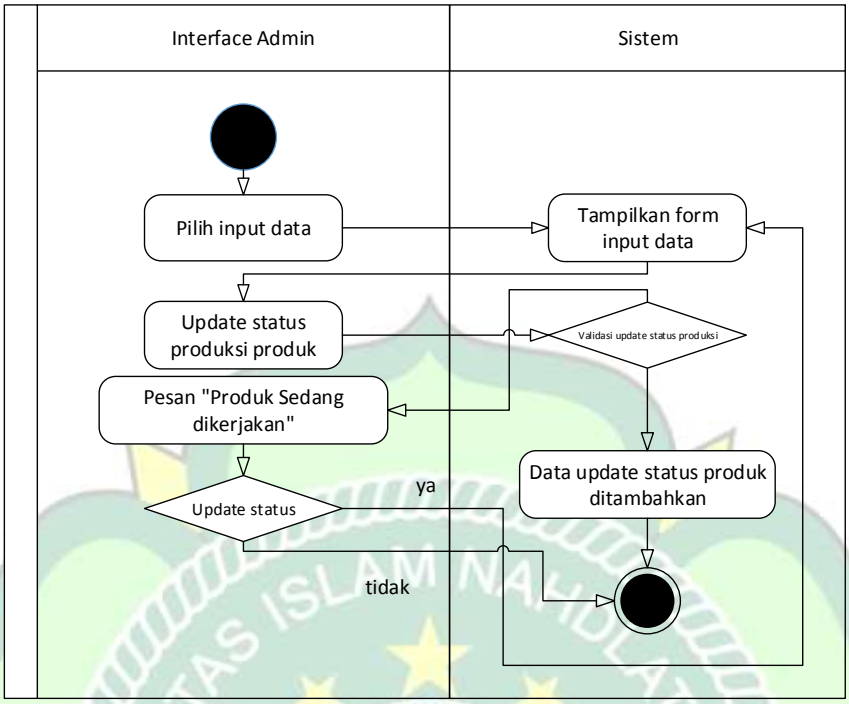
Gambar di atas adalah contoh alur dari *use case diagram* yang merupakan sebuah proses analisa untuk memahami bagaimana sistem bekerja dari pembeli yang ingin mencari produk serta proses transaksi ke penjualnya.

2.2.7 Activity Diagram

Activity Diagram merupakan gambaran perilaku alur kerja sistem untuk aktifitas, objek, *state*, transisi *state*, dan *event*. *Activity Diagram* berfokus pada aktifitas yang terjadi dalam proses tunggal suatu sistem dan menunjukkan ketergantungan terhadap setiap aktifitasnya[10]. Berikut adalah bentuk simbol yang digunakan dalam penggambaran *Activity Diagram* :

Tabel 2.3 Simbol *Activity Diagram*

No	Simbol	Nama	Keterangan
1.		Start point	Merupakan awal dari sebuah aktivitas.
2.		End point	Merupakan Akhir dari sebuah aktivitas.
3.		Activities	Menggambarkan suatu proses atau kegiatan.
4.		Fork (percabangan)	Merupakan symbol yang digunakan untuk menggambarkan kegiatan yang dilakukan secara pararel menjadi satu.
5.		Join (penggabungan)	Digunakan untuk menunjukkan adanya dekomposisi.
6.		Decision point	Digunakan untuk menentukan pilihan *True or False*/Benar atau Salah.
7.		Swimlane	Digunakan untuk membagi activity diagram.



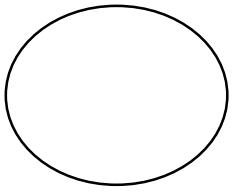
Gambar 2.5 Contoh *Activity Diagram*

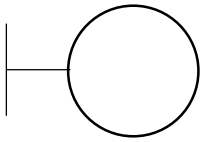
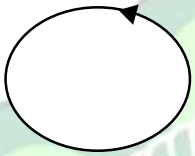
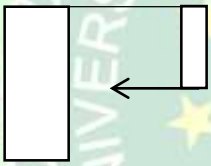
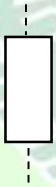
Gambar di atas adalah alur kerja dari sistem dan menunjukkan ketergantungan pada setiap aktifitasnya seperti cara proses input data untuk mencari stok produk.

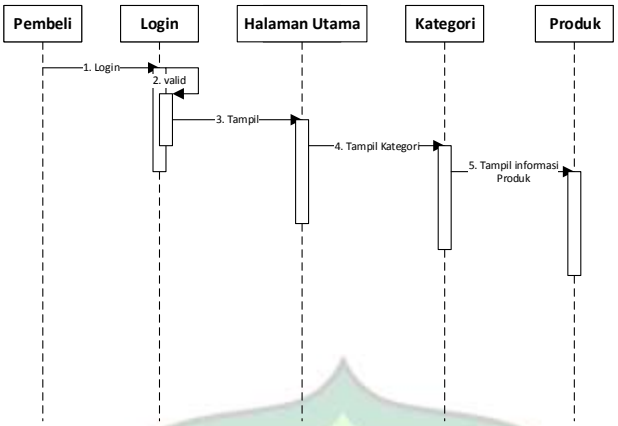
2.2.8 Sequence Diagram

Sequence Diagram merupakan diagram interaksi objek yang disusun berdasarkan urutan waktu yang memeberikan gambaran tahap demi tahap sesuai dengan skenario, termasuk kronologi kejadian yang seharusnya dilakukan secara logis untuk menghasilkan keluaran yang sesuai dengan *Use Case Diagram*.[10] Berikut adalah bentuk simbol yang digunakan dalam penggambaran *Sequence Diagram* :

Tabel 2.4 Simbol *Sequence Diagram*

No	Simbol	Nama	Keterangan
1.		Entity class	Merupakan bagian dari sistem yag berisi kumpulan kelas berupa entitas yang membentuk gambaransistem dan

			menjadi acuan penyusunan basis data.
2.		Boundary class	Merupakan kumpulan class yang menjadi interface atau interaksi antara satu actor dengan sistem seperti tampilan form.
3.		Control class	Merupakan objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab terhadap entitas.
4.		Message	Simbol mengirim pesan antar class.
5.		Recursive	Menggambarkan pengiriman pesan kepada dirinya sendiri.
6.		Activation	Mewakili sebuah eksekusi operasi dari objek, durasi aktivitas operasi berbanding lurus dengan panjang kotak.
7.		Lifeline	Garis titik-titik yang terhubung dengan objek, sepanjang lifeline terdapat actovation.

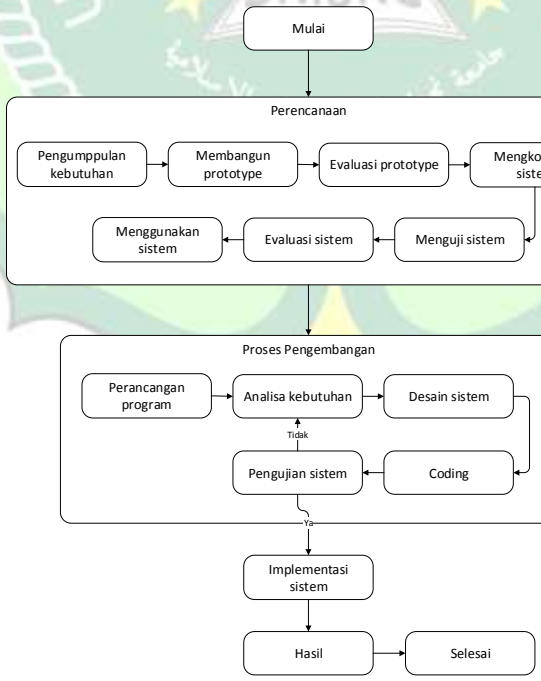


Gambar 2.6 Contoh *Sequence Diagram*

Gambar di atas adalah alur interaksi berdasarkan urutan waktu termasuk kronologi kejadian untuk menghasilkan *output* yang sesuai *use case diagram*.

2.3 Kerangka Pemikiran

Pada tahap ini peneliti membuat suatu kerangka pemikiran yang menjadi gambaran dari penelitian dari awal sampai selesai melakukan penelitian dalam menjelaskan konsep hingga memberikan pandangan terhadap penelitian yang akan dilakukan.



Gambar 2.7 Kerangka Berpikir