**BAB II**

# LANDASAN TEORI

## Teori-teori Umum

* 1. Data

Menurut Sopingi (2015:20):

”Data merupakan kumpulan fakta atau angka atau segala sesuatu yang dapat dipercaya kebenarannya, dan karenanya dapat dijadikan sebagai dasar untuk menarik suatu kesimpulan. Data dapat berupa angka atau bilangan, dan biasanya disebut sebagai data kuantitatif. Data dapat juga berupa konsep atau kategori yang bukan berupa angka, dan biasanya disebut sebagai data kualitatif.”

Menurut Tyoso (2016:40):

”Data informasi sering diberikan pengertian yang sama walaupun ada kecenderungan anggapan bahwa data adalah fakta yang belum dievaluasi sebagai bahan mentah informasi. Informasi dianggap sebagai data yang diproses ke dalam beberapa bentuk tertentu yang mempunyai makna. Data diartikan pula apa pun yang sudah tertentu, tidak menjadi masalah apa pun bentuknya, dari mana asalnya, sehingga muncul pengertian informasi adalah suatu jenis data atau data adalah suatu tipe tertentu informasi.”

Dari definisi-definisi tersebut, dapat disimpulkan bahwa data dan informasi itu berbeda. Data merupakan kumpulan fakta atau angka atau segala sesuatu yang dapat dipercaya kebenarannya.

* 1. Informasi

Menurut Djahir (2014:8):

“Informasi merupakan hasil dari pengolahan data menjadi bentuk yang lebih berguna bagi yang menerimanya yang menggambarkan suatu kejadian-kejadian nyata dan dapat digunakan sebagai alat bantu untuk pengambilan suatu keputusan.”

Menurut Tyoso (2016:21):

”Informasi adalah suatu pertambahan dalam ilmu pengetahuan yang menyumbangkan kepada konsep kerangka kerja yang umum dan fakta-fakta yang diketahui. Informasi bertumpu pada konteks dan pengetahuan umum si penerima untuk kepentingannya. Agar dapat menerima informasi dengan sebenarnya, pada contoh di atas, anda tidak hanya ingin tahu “lima” berarti lima ribu rupiah, tetapi anda juga ingin tahu penjualan sebelumnya dari divisi tersebut dan perkiraan”

Menurut kedua definisi di atas, Informasi adalah data yang sudah diolah sehingga mempunyai arti yang penting sesuai konteks yang familiar dengan pihak penerima informasi.

* 1. Jaringan Komputer (Internet)

Menurut Wahana Komputer (2010:2):

“Jaringan komputer adalah sistem yang terdiri dari komputer-komputer serta piranti-piranti yang saling terhubung sebagai satu kesatuan. dengan dihubungkannya piranti-piranti tersebut, alhasil dapat saling berbagi sumber daya antara satu pihak dengan piranti lainnya.Dalam istilah komputer, jaringan merupakan penghubung antara dua komputer atau lebih yang tujuannya utamanya adalah berbagi data. Tujuan dari jaringan komputer adalah membagi sumber daya, contohnya berbagai pemakaian *printer, CPU, memory, harddisk,* untuk komunikasi, contohnya surat elektronik, *instant messaging, chatting,* dan akses informasi, contohnya *web browsing* agar dapat mencapai tujuan yang sama setiap bagian dari jaringan komputer meminta dan memberikan layanan (*service*). Pihak yang Meminta layanan disebut *client* dan yang memberikan layanan disebut server. arsitektur ini disebut dengan sistem *client-server* dan digunakan pada hampir seluruh jaringan komputer.”

Menurut Krisianto (2014:1):

“Secara mudahnya, internet adalah salah satu bentuk media komunikasi dan informasi interaktif. Wujud internet adalah jaringan komputer yang terhubung di seluruh dunia. Internet digunakan untuk mengirim informasi antara komputer di seluruh dunia sehingga, melalui internet kita bisa mengakses dan bertukar informasi secara cepat. Mengakses internet berarti mengakses informasi. Alat untuk mengakses internet bisa menggunakan komputer/laptop, *handphone*, dan berbagai *smartphone*/*gadget*. Internet menawarkan kecepatan informasi yang tidak terbayangkan sebelumnya. Dalam hitungan detik, kita bisa saling bertukar informasi dengan seseorang di belahan dunia yang lain.”

Berdasarkan dua definisi tersebut, dapat disimpulkan bahwa Jaringan Komputer (*Internet*) adalah sistem yang terdiri dari komputer-komputer serta piranti-piranti yang saling terhubung sebagai satu kesatuan dan dapat saling berbagi sumber daya dan informasi antara satu pihak dengan piranti lainnya dengan sangat cepat.

* 1. Rekayasa Perangka Lunak (*Software*)

Menurut Supriyanto dan Musin (2012:40):

“Rekayasa perangkat lunak adalah sebuah profesi yang dilakukan oleh seorang perekayasa perangkat lunak yang berkaitan dengan pembuatan dan pemeliharaan aplikasi perangkat lunak dengan menerapkan teknologi dan praktik dari ilmu komputer, manajemen proyek, dan bidang-bidang lainnya. Perangkat lunak adalah instruksi langsung komputer untuk melakukan pekerjaan dan dapat ditemukan di setiap aspek kehidupan modern dari aplikasi yang kritis untuk hidup (*life-critical*), seperti perangkat pemantauan medis dan pembangkit tenaga listrik sampai perangkat hiburan, seperti *video game*. Banyak produk perangkat lunak berisi jutaan baris kode yang diharapkan dapat melakukan pekerjaan dengan baik dalam menghadapi perubahan kondisi. Semua perangkat lunak juga membutuhkan keandalan yang tinggi dan harus dihasilkan secara ekonomis. Teknik rekayasa perangkat lunak akan meningkatkan fungsionalitas dan efisiensi aplikasi dan juga kemudahan dan efisiensi dari pengembang perangkat lunak.“

Menurut Simarmata (2010:40):

“Perangkat lunak adalah istilah yang digunakan untuk menggambarkan instruksi-instruksi yang memberitahu perangkat keras untuk melakukan suatu tugas sesuatu dengan perintah. Tanpa perangkat lunak, perangkat keras tidak ada gunanya.”

Sehingga dapat disimpulkan bahwa pembuatan dan pemeliharaan aplikasi perangkat lunak atau instruksi langsung komputer untuk melakukan pekerjaan dengan menerapkan teknologi dan praktik dari ilmu komputer, manajemen proyek, dan bidang-bidang lainnya.

* 1. Sistem

Menurut Benjamin S Blanchard dan Wolter J Fabrycky (2010:1):

*“A system is complete set of component which interact or are independent from on stage to another; for any specific task performance or generating required output. Every system is structured such that its components are related. Also any structure show behavior, which translates the input given to it into organized output.”*

“Suatu sistem adalah kumpulan komponen lengkap yang berinteraksi atau independen dari tahap satu ke yang lain, untuk kinerja pekerjaan tertentu atau untuk menghasilkan *output* yang diperlukan. Setiap sistem terstruktur sedemikian rupa sehingga komponen-komponennya saling terkait. Dan juga semua jenis struktur memiliki karakter tersendiri, yang mana dapat di terjemahkan untuk perintah yang di *input* kepada/menjadi *output* yg telah diatur.”

Menurut Muhammad Muslihudin dan Oktafianto (2016:2):

“Sistem adalah sekumpulan komponen atau jaringan kerja dari prosedur-prosedur yang saling berkaitan dan saling bekerja sama membentuk suatu jaringan kerja untuk mencapai sasaran atau tujuan tertentu”

Menurut Tyoso (2016:1):

“Sistem merupakan suatu kumpulan dari komponen-komponen yang membentuk satu kesatuan. Sebuah organisasi dan sistem informasi adalah sistem fisik dan sosial yang ditata sedemikian rupa untuk mencapai tujuan tertentu. Seorang manajer yang mengorganisasikan pekerjaan unitnya adalah orang yang menciptakan sesuatu sistem pengembangan *personal computer* (PC) menciptakan suatu sistem yang terdiri dari perangkat keras dan lunak PC ini menjadi *subsistem* dari suatu perangkat konferensi elektronik. Sistem fisik dan sosial adalah sistem yang abstrak *(abstract system)* dari konsep dan ide.”

Menurut Mulyani (2014:4-11):

Sistem mempunyai komponen-komponen yang membentuk sistem dan saling bekerja sama. Salah satu dari komponen ini tidak boleh hilang ataupun rusak. Jika salah satu dari komponen ini tidak bekerja, kerja sistem akan terhenti dan tujuan dari sistem tidak pernah akan tercapai. Komponen-komponen biasanya berupa subsistem baik berupa fisik maupun abstrak. Subsistem adalah sebuah sistem yang lebih kecil dari sistem yang menjadi lingkungannya.

Dalam melakukan pekerjaannya, komponen-komponen dalam sistem harus saling terintegrasi satu sama lain. Sistem mempunyai batasan sistem, yaitu sebuah batasan-batasan yang bisa memberikan gambaran pemisah antara lingkup sistem dengan batas luar sistem. Tujuan sistem merupakan target atau hasil akhir yang sudah dirancang oleh pembuat sistem.

Dapat disimpulkan bahwa sistem adalah sekumpulan komponen atau jaringan kerja dari prosedur-prosedur yang saling berkaitan dan saling bekerja sama membentuk suatu jaringan kerja untuk mencapai sasaran atau tujuan tertentu di mana sistem memiliki *input*, proses, dan *output* untuk mengerjakan , menyelesaikan pekerjaannya dan mencapai tujuannya.

* 1. Statistika

Menurut Winarno (2014:2):

“Statistika adalah bidang yang berkaitan dengan pengambilan keputusan yang berkaitan dengan angka. Dengan statistik, para ahli statistik berusaha mencari jawaban dari data-data berupa angka. Statistika merupakan metode yang membuat data-data yang sangat banyak bisa dipahami dan diketahui artinya. Statistika merupakan ilmu yang berkenaan dengan data, sedang statistik adalah data, informasi, atau hasil penerapan algoritma statistika pada suatu data.”

Menurut Furqon (2014:3):

“Statistika adalah bagian dari matematika yang secara khusus membicarakan cara-cara pengumpulan, pengolahan, penyajian, analisis dan penafsiran data. Dengan kata lain, istilah statistika di sini digunakan untuk menunjukkan tubuh pengetahuan (*body of knowledge*) tentang cara-cara penarikan sampel (pengumpulan data), serta analisis dan penafsiran data.”

Dari beberapa definisi tersebut, dapat disimpulkan bahwa statistika bagian dari matematika yang secara khusus membicarakan cara-cara pengumpulan, pengolahan, penyajian, analisis dan penafsiran data. Statistika merupakan metode yang membuat data-data yang sangat banyak bisa dipahami dan diketahui artinya.

1. **Teori-teori Khusus**
   1. Cloud Computing

Menurut National Institute of Standards and Technology- Dijiang Huang, Huijun Wu (2017:6)

*“Cloud computing is a model for enabling ubiquitous, convenient, on- demand network access to a shared pool of configurable computing resources(e.g., network, servers, storage , applications, and services) that can be rapidly provisioned and released with minimal management affort or service provider interaction.”*

Komputasi awan adalah sebuah model untuk memngkinkan akses jaringan dimana saja, mudah dan sesuai dengan permintaan akses ke kumpulan sumberdaya yang dapat dikonfigurasi bersama (misalnya jaringan, server, penyimpanan, aplikasi, dan layanan) yang dapat diterapkan secara tepat dan dirilis dengan upaya manajemen atau interaksi layanan jasa yang minimal.

Menurut Mohamad Jamil, Roshihan, dan Achmad Fuad(2016:3):

“Cloud Computing merupakan evolusi yang mengadopsi virtualisasi, arsitektur servis oriented dan utility computing. Cloud computing memungkinkan konsumen teknologi untuk memikirkan komputasi secara efektif dengan biaya minimal dan dapat diandalkan. Hal lain yang juga tidak perlu lagi dikhawatirkan oleh pengguna adalah bagaimana membangunnya, cara kerjanya, siapa yang mengoperasikann atau dimana harus diletakkan.”

Menurut definisi-definisi tersebut, bahwa komputasi awan atau cloud computing merupakan pemanfaatan teknologi computer berbasiskan jaringan internet yang meliputi penyimpanan, server, jaringan, serta aplikasi, dan sebagainya.

* 1. Mobile Computing

Menurut National Institute of Standards and Technology- Dijiang Huang, Huijun Wu (2017:9)

*“Mobile computing focuses on device mobility and context awareness considering networking and mabile resource/ data access. Mobile computing applications usually rely on mobile device create, access, process, store, and communicate information without being constrained to a single location. Mobile computing usually considers similar type of mobile devices that offers may otherwise unattainable benefits to organization that choose to integrate it into their fixed information system.”*

Komputasi *mobile* berfokus pada *device* yang dapat bergerak dan kontek kesadaran yang mempertimbangkan jaringan dan sumberdaya mobile atau data akses. Aplikasi komputasi mobile biasanya tergantung pada pembuatan perangkat mobile, akses, proses, penyimpanan, dan menyampaikan informasi tanpa terbatas pada satu lokasi. komputasi mobile biasanya mempertimbangkan kesamaan jenis dari perangkat mobile yang menawarkan manfaat lain yang mungkin tak terjangkau bagi organisasi yang memilih untuk mengintegrasikannya ke dalam sistem informasi tetap mereka.

Menurut Asoke K. Talukdar (2010:5)

*“Mobile computing can be defined as a computing environment over physical mobility. The user of mobile computing environment will be able to access data information or other logical object from any device in any network while on the move. Mobile computing system allows a user to perform a task from anywhere using a computing device in public (the web), corporate (business information) and personal information spaces medical record, address book). While on the move, preferred device will be a mobile device, while back at home or in the office the device could be a desktop computer. To make the mobile computing environment ubiquitous, it is necessary that the communication bearer is spread over both wired and wireless media.”*

Komputasi *mobile* dapat didefinisikan sebagai sebuah lingkungan komputasi lebih dari mobilitas fisik. ruang lingkup *user* dari *mobile computing* akan dapat mengakses data informasi dari logikal objek yang lain dari semua perangkat di semua jaringan ketika sedang bergerak. sistem *mobile computing* memungkinkan *user* melakukan pekerjaan dari mana saja dengan menggunakan perangkat komputasi publik (web), perusahaan (informasi bisnis) dan ruang informasi pribadi rekam medis, buku alamat. ketika bergerak, perangkat utama akan menjadi perangkat bergerak, ketika kembali dirumah atau kantor maka akan menjadi dekstop komputer. Untuk membuat lingkungan komputasi *mobile* di mana-mana, membutuhkan komunikasi bawaan yang tersebar di kedua media kabel dan nirkabel.

Sehingga dapat disimpulkan bahwa komputasi *mobile* merupakan teknologi komputasi yang memungkinkan *user* melakukan pekerjaannya meskipun sedang bergerak memanfaatkan jaringan nirkabel untuk mengakses kebutuhan informasi pada sebuah perangkat *mobile*.

* 1. UML

Menurut Verdi (2012:194):

“*Unified Modelling Language* (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.”

Menurut Mulyani (2016:42-60):

*UML* memiliki banyak diagram yang digunakan untuk melakukan pemodelan data maupun sistem, beberapa diantaranya yaitu *Use Case Model* (memiliki 3 diagram) dan *Activity Diagram*

1. Use Case Model

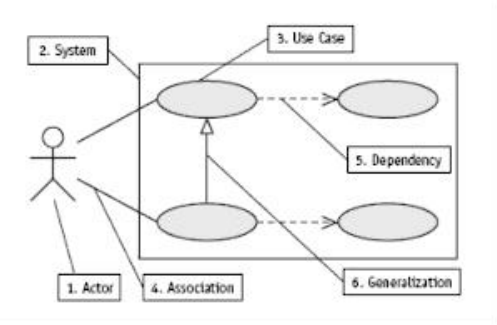
*Use Case Model* yaitu sekumpulan diagram yang Digunakan untuk menggambarkan sistem ke dalam notasi grafis. Use Case Model lebih terfokus pada penggambaran tujuan dari suatu sistem. Use Case Model terdiri dari tiga notasi yaitu:

1. Use Case Diagram

Use Case Diagram, yaitu diagram yang digunakan untuk menggambarkan hubungan antara sistem dengan aktor. diagram ini hanya menggambarkan secara global. Karena Use Case Diagram yang menggambarkan sistem secara *global*, maka elemen-elemen yang digunakan pun sangat sedikit, berikut ini elemen-elemen yang digunakan pada *use case diagram.*

**GAMBAR 2.1**

ELEMEN DARI USE CASE DIAGRAM

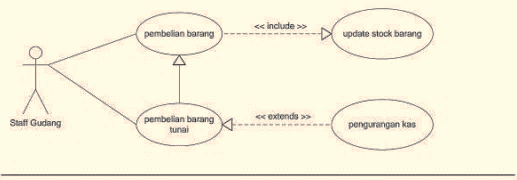
**

*Sumber: Mulyani, (2016:43)*

* *Sistem*, merupakan batasan-batasan proses yang sudah kita deskripsikan dalam sebuah sistem.
* Aktor, elemen yang menjadi pemicu sistem. Aktor bisa berupa orang, mesin ataupun sistem lain yang berinteraksi dengan *use case*.
* *Use case*, potongan proses yang merupakan bagian dari sistem.
* *Association*, Menggambarkan interaksi antara *use case* dan aktor.
* *Dependency*, Menggambarkan relasi (relationship) antara dua *use case.* Ada 2 (dua) tipe dari *dependency* yaitu, *include* dan *extends.* Include merupakan tipe dari *dependency* yang menghubungkan dua *use case* dimana, satu use case membutuhkan use case yang satunya sedangkan extends adalah tipe dadri dependency yang mehbugunkan dua use case dimana satu use case terkadang akan memanggil use case yang satunya, tergantung pada kondisi.
* *Generalization,* menggambarkan pewarisan antara dua aktor atau use case dimana salah satu aktor atau use case mewarisi properties ke aktor atau use case yang satunya.

**GAMBAR 2.2**

CONTOH USE CASE DIAGRAM



*Sumber: Mulyani, (2016:44)*

1. Use Case Narrative

*Use Case Narrative,* Yaitu uraian deskripsi dari *use case diagram* sehingga pengguna *UML* bisa mengetahui detail dari proses yang ada *pada use case diagram*. Berikut ini yang elemen-elemen yang digunakan pada *use case narrative*.

1. *Assumptions*, yaitu asumsi-asumsi yang harus bernilai *true*, agar *use case* bisa bekerja. Assumptions identik dengan batasan dari use case.
2. *Pre-Conditions*, yaitu indentik dengan a*ssumptions* hanya saja *pre-condition* harus sudah diperiksa sebelum *use case* melakukan pekerjaannya agar kondisi bernilai *true*. seperti misalnya seorang programmer yang membuat sebuah fungsi dengan sebuah *parameter,* sebelum fungsi itu memproses *parameter* yang diinputkan, *parameter* tersebut diperiksa dulu sehingga memungkinkan kondisinya bernilai *true.*
3. *Use case initiation*, yaitu *trigger* (pemicu) sehingga *use case* mulai bekerja.
4. *Process or dialog*, yaitu urutan langkah (*step by step)*  deskripsi dari *use case.*
5. *Use case termination*, yaitu hal yang membuat *use case* berhenti bekerja.
6. *Post-conditions*, yaitu ansumsi-ansumsi yang harus bernilai *true,* ketika *use case* berhenti bekerja.

**TABEL 2.1**

CONTOH USE CASE NARRATIVE DARI

CONTOH SEBELUMNYA

|  |  |
| --- | --- |
| *Fields name* | *Description* |
| *Assumptions* | Aktor mempunyai *authority* untuk melakukan pembelian barang |
| *Pre-conditions* | 1. Stok barang di bawah atau sama dengan jumlah minimum 2. *Approve* oleh kepala gudang pembelian |
| *Use case initiation* | Pemeriksaan stok barang. |
| *Dialog* | 1. *Input* id barang 2. Periksa stok barang 3. Aktor melakukan pengajuan pembelian barang 4. Menghubungi supplier 5. Melakukan pembelian barang |
| *Use case termination* | 1. Aktor membatalkan pembelian barang 2. Pembelian barang selesai dilakukan |
| *Post-conditions* | *Normal termination*:   1. Stok barang di gudang bertambah 2. Adanya pengurangan kas untuk pembelian tunai   *Cancel termination*:   1. Stok barang tetap. |

*Sumber: Mulyani, (2016:45-46)*

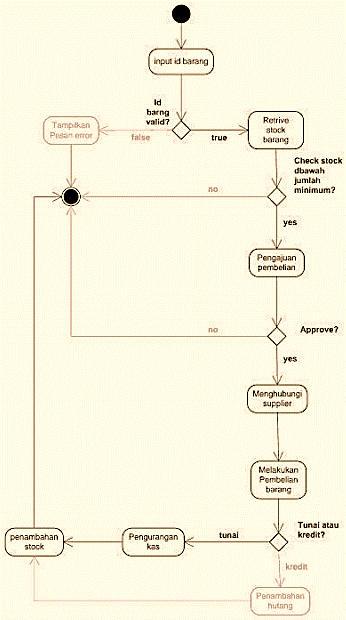
1. Use Case Scenario

*Use Case Scenario*, yaitu diagram yang menggambarkan logika-logika (kemungkinan skenario) dari *use case narrative*. Disini kita akan memecah beberapa kemungkinan kejadian (skenario) pada *use case narrative* menjadi beberapa skenario. Untuk lebih jelasnya, dibawah ini kita akan mencoba membuat *use case scenario* dengan menggunakan *use case diagram* dan *use case narrative* yang sudah kita bahas diatas

**GAMBAR 2.3**

CONTOH USE CASE SCRENARIO

SELURUH KEMUNGKINAN



*Sumber: Mulyani, (2016:47)*

1. Activity Diagram

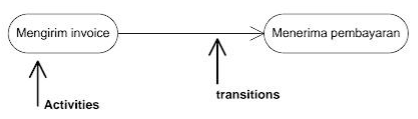
*Activity Diagram*, yaitu *diagram* yang digunakan untuk menggambarkan alur kerja (aktivitas) pada *use case* (proses), logika, proses bisnis dan hubungan antara aktor dengan alur-alur kerja *use case.* Jika Anda sudah terbiasa dengan *flowchart*, maka Anda tidak akan merasa kesulitan dalam mempelajari *activity diagram*, karena *activity diagram* identik dengan *flowchart*, hanya saja ada beberapa notasi tambahan yang digunakan untuk kasus-kasus tertentu. Berikut ini dijelaskan elemen-elemen dari *activity diagram*

* 1. *Activities*, yaitu elemen yang digunakan untuk menggambarkan aktivitas.
  2. *Transitions*, yaitu elemen yang digunakan untuk menggambarkan transisi dari elemen yang satu ke elemen yang lainnya

**GAMBAR 2.4**

CONTOH ELEMEN ACTIVITIES

DAN TRANSITIONS

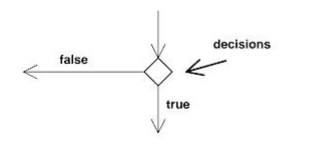


*Sumber: Mulyani, (2016:55)*

* 1. *Decisions*, yaitu elemen yang digunakan untuk percabangan logika. Elemen ini sering kita jumpai pada *flowchart* terutama *flowchart* yang digunakan untuk menggambarkan algoritma.

**GAMBAR 2.5**

CONTOH ELEMEN DECISIONS

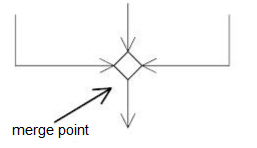


*Sumber: Mulyani, (2016:56)*

* 1. *Merge Point,* yaitu elemen yang digunakan untuk menggabungkan percabangan proses. Elemen ini merupakan kebalikan dari elemen *decisions*, dimana jika *decisions* digunakan untuk percabangan, sedangkan *merge point* digunakan sebagai penggabungan dari percabangan.

**GAMBAR 2.6**

CONTOH NOTASI MERGE POINT

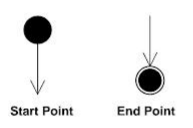


*Sumber: Mulyani, dengan pengubahan (2016:56)*

* 1. *Start Point,* yaitu elemen yang digunakan untuk memulai *activity diagram.*
  2. *End Point,* yaitu elemen yang digunakan untuk mengakhiri *activity diagram.*

**GAMBAR 2.7**

CONTOH START POINT DAN END POINT

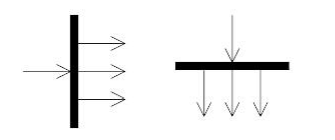


*Sumber: Mulyani, (2016:57)*

* 1. *Concurrency,* yaitu elemen yang digunakan sebagai percabangan proses (bukan percabangan logika). Proses yang ada di dalam elemen ini, bisa dilakukan secara *random* (tidak berurutan).

**GAMBAR 2.8**

CONTOH NOTASI CONCURRENCY

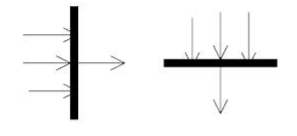


*Sumber: Mulyani, (2016:57)*

* 1. *Synchronization,* yaitu elemen yang digunakan untuk menggabungkan proses yang dipisahkan oleh *concurrency.*

**GAMBAR 2.9**

CONTOH ELEMEN SYNCHRONIZATION

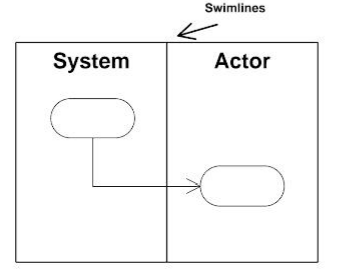


*Sumber: Mulyani, (2016:58)*

* 1. *Swimlines,* yaitu elemen yang digunakan untuk memisahkan antara aktor dan sistem ataupun antara aktor yang satu dengan aktor yang lain atau antara sistem yang satu dengan sistem yang lain.

**GAMBAR 2.10**

CONTOH ELEMEN SWIMLINES



*Sumber: Mulyani, (2016:58)*

* 1. Sinyal, yaitu acuan waktu yang bisa dijadikan *trigger* (pemicu) untuk aktivitas tertentu, misalnya setiap akhir jam kerja seluruh staf wajib memberikan laporan kepada manajer. Setiap akhir jam kerja bisa kita simbolkan dengan sinyal waktu.

**GAMBAR 2.11**

CONTOH ELEMEN SINYAL

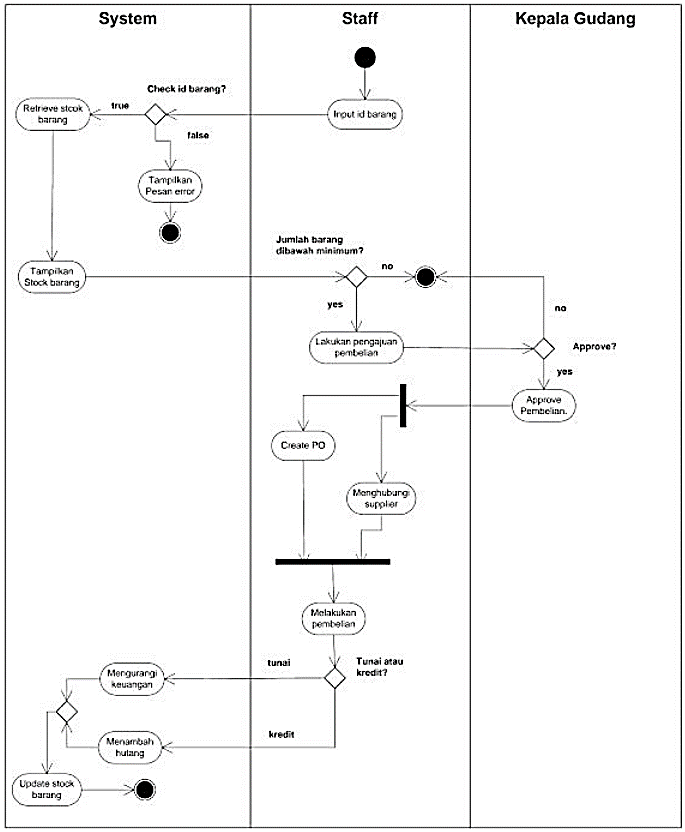


*Sumber: Mulyani, (2016:59)*

Berikut ini contoh kasus dari pengunaan *activity diagram.* Kasus yang diambil seputar pembelian barang yang sudah kita buatkan *use case model* nya di atas

**GAMBAR 2.12**

CONTOH ACTIVITY DIAGRAM



*Sumber: Mulyani, (2016:60)*

Berdasarkan pendapat dan uraian sebelumnya, *Unified Modelling Language* (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem piranti lunak.

* 1. Firebase

Menurut Mayur Tanna dan Harmeet Singh (2018:11):

“*Firebase is a mobile and web application development platform becked by google. It is a one-stop solution for all your need to develop high quality mobile and web applications. It includes various products, such as realtime Database, Crash reporting, Cloud Firestore, Cloud storage, cloud functions, authentication, Hosting, test lab for Android, and performance monitoring for iOS, which can be used to develop and test Realtime applications by focusing on the user’s needs, rather than the technical complexities*.”

Firebase merupakan sebuah *platform development* mobile dan web aplikasi buatan google. Ini merupakan sebuah solusi untuk menangani semua yang dibutuhkan untuk membuat aplikasi mobile dan web yang berkualitas tinggi. Firebase mencakup variasi produk antara lain, *Realtime* *Database*, *Crash Reporting*, *Cloud firestore*, Cloud storage, cloud functions, authentication, Hosting, test lab untuk Android dan performa monitoring untuk iOS, dimana dapat dipakai untuk membuat dan tes aplikasi Realtime dengan memfokuskan kebutuhan dari *user*, melebihi kompleksitas teknis.

Menurut Bill Stonehem (2016:1) :

“*Android Firebase is a cloud service provider as well as a backend business that allows you to obtain organized data for mobile apps. This is an important aspect as almost all mobile apps today needs user verification and update. Firebase is easy to use and allows quick reading and writing of data event for beginners.”*

Android Firebase merupakan penyedia layanan *cloud*  serta bisnis *backend* yang memungkinkan anda untuk memperoleh data yang terorganisir untuk aplikasi perangkat seluler. Ini merupakan aspek penting sebagai hamper semua aplikasi perangkat seluler sekarang ini membutuhkan verifikasi pengguna dan juga *update*. Firebase mudah untuk digunakan dan memungkinkan pembacaan yang cepat serta penulisan data walaupun untuk pengguna baru.

Menurut Scott Domes ( 2017:73):

*“Firebase is a project by google design to help developers build apps without worrying about backend infrastructure. It operates on the freemium model, based on the amount of requests your backend has to respond to as well as the amount of storage you need. For our purposes, it the perfect fit for quickly developing a small prototype. Firebase is a database, a hosting solution, and built-in authentication. Beyond that, it also offers something called cloud function, which are snippets of cod automatically run in response of certain event.”*

Firebase adalah sebuah proyek dari *google design* untuk membantu pengembang membangun aplikasi tanpa harus mengkhawatirkan infrastruktur *backend*. Ini beroperasi pada model freemium, berdasarkan pada jumlah permintaan yang harus di-*respond* oleh *backend*  anda, serta jumlah penyimpanan yang anda butuhkan. Tujuannya kebutuhan untuk mengembangkan *prototype* kecil dengan cepat. Firebase adalah basis data, solusi hosting, dan autentikasi bawaan. Selain itu, juga menawarkan sesuatu yang disebut fungsi cloud, yang merupakan cuplikan cod yang secara otomatis dijalankan sebagai respons terhadap peristiwa tertentu.

Menurut Laurence Moroney ( 2017:1)

*“it’s not a replacement for you exiting APIs for building android, iOS, or Web apps. It’s backand enhancement, giving you common services that you might need . such as a database backend, secure authentication, messaging, and more. This saves you the need to build them yourself, allowing you to focus on what makes your app distinct. Additionally, it has through referrals, linking, and more.”*

Ini bukan merupakan pengganti APIs yang telah ada untuk mengembangkan android, iOS, atau Web Aplikasi. Ini merupakan bantuan backend, yang memberikan anda layanan umum yang mungkin dibutuhkan. sebagai sebuah *database* *backend*, otentikasi keamanan, pesan, dan lainnya. Ini menyelamatkan anda dari harus membuatnya sendiri, sehingga memungkinkan anda untuk lebih fokus pada pembuatan aplikasi. Selain itu juga memiliki referensi, menghubungkan dan sebagainya.

Sehingga dari sana ditarik kesimpulan, Firebase merupakan sebuah layanan *backend* yang dilengkapi dengan *realtime* *database*, otentifikasi keamanan, pesan juga mencakup layanan penyimpanan berbasis *cloud*.

* 1. Java

Menurut Nofriandi (2015:1):

”Bahasa pemrograman *Java* merupakan salah satu dari sekian 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 Microsystems.* Bahasa pemrograman ini merupakan pengembangan dari bahasa pemrograman C++ karena banyak mengadopsi sintak C dan C++. Saat ini *Java* merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi atau pun aplikasi berbasis *web*.

Kelebihan *Java* dari bahasa pemrograman yang lain adalah bisa dijalanakan di berbagai jenis sistem operasi sehingga dikenal juga bahasa pemrograman multi platform, bersifat pemrograman berorientasi *object* (PBO), memiliki *library* yang lengkap.”

Menurut Mardiani, et al (2017:29):

“Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus pada sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat *open source*.”

Java adalah bahasa pemrograman berorientasi objek (OOP) yang dapat dijalankan di berbagai sistem operasi termasuk telepon genggam dan bersifat *open source*.

* 1. Android

Menurut Winarno dan Ali (2012:6):

“*Android* merupakan sistem operasi yang dikembangkan oleh *Google*. Sistem operasi yang berbasis *Linux* dan dapat digunakan untuk ponsel. *Android* menyediakan platform terbuka bagi para developer sehingga dapat membuat aplikasi sendiri yang dapat dijalankan di *smartphone.”*

Menurut Griffiths dan Griffiths (2015:2)

*“Android is the world’s most popular mobile platform. At the last count, there were over one billion active Android devices worldwide, and that number is growing rapidly. Android is a comprehensive open source platform based on Linux and championed by Google. It’s a powerful development framework that includes everything you need to build great apps using a mix of Java and XML. What’s more, it enables you to deploy those apps to a wide variety of devices—phones, tablets and more. Layouts define what each screen looks like. A typical Android app is comprised of one or more screens. You define what each screen looks like using a layout to define its appearance. Layouts are usually defined using XML, and can include GUI components such as buttons, text fields, and labels.*

*Java code defines what the app should do. Layouts only define the appearance of the app. You define what the app does by writing Java code. A special Java class called an activity decides which layout to use and tells the app how to respond to the user. As an example, if a layout includes a button, you need to write Java code in the activity to define what the button should do when you press it.”*

(*Android* adalah *platform* mobile terpopuler di dunia. Pada hitungan terakhir, ada lebih dari satu miliar perangkat Android aktif di seluruh dunia, dan jumlah tersebut berkembang dengan cepat. Android adalah platform *open source* yang komprehensif di *Linux* dan diperjuangkan oleh *Google*. Android adalah kerangka pembangunan yang kuat yang mencakup semua yang anda butuhkan untuk membangun aplikasi hebat menggunakan gabungan *Java* dan *XML*. Apa lebih, ini memungkinkan anda untuk menyebarkan aplikasi tersebut ke berbagai macam perangkat – telepon, tablet dan banyak lagi.

*Layout-layout* menentukan seperti apa tampilan masing-masing layar. Aplikasi *Android* yang khas terdiri dari satu atau lebih layar. Anda yang menentukan apa yang setiap layar tampilkan seperti menggunakan *Layout* untuk menentukan tampilannya. *Layout* biasanya didefinisikan menggunakan *XML*, dan dapat mencakup komponen *GUI* seperti *button, text field,* dan *label.*

Kode *Java* mendefinisikan apa yang harus aplikasi lakukan. *Layout* hanya menentukan tampilan aplikasi. Anda mendefinisikan apa yang aplikasi lakukan dengan menulis kode *Java*. Sebuah *Class Java* khusus yang disebut dengan *Activity* menentukan *Layout* yang akan digunakan dan memberi tahu aplikasi cara merespons ke *user*. Sebagai contoh, jika *Layout* menyertakan tombol, anda perlu menulis kode *Java* dalam aktivitas untuk menentukan apa yang harus dilakukan tombol ketika anda menekannya.)

Jadi dapat disimpulkan bahwa *Android* merupakan sistem operasi berbasis *Linux* dan dapat digunakan untuk ponsel yang dikembangkan oleh *Google. Android* menggunakan gabungan antara *Java dan XML.*