PROPOSAL PENELITIAN

# Analisa Simulasi Monte Carlo Dalam Menentukan Pendapatanpenjualankeripik Maco Badarai Istiqomah Padang Sumatera Barat



# OLEH :

**M. HAFIZH 1023129201**

# PROGRAM STUDI TEKNIK INFORMATIKA FAKULTAS ILMU KOMPUTER

**UNIVERSITAS PUTRA INDONESIA “YPTK” PADANG**

# ABSTRAK

Perancangan sistem simulasi menentukan pendapatan keripik maco dengan menggunakan metode *monte carlo* pada *home-industry* Istiqomah bertujuan untuk memprediksi penjualan setiap produk barang untuk bulan berikutnya dan mencari pendapatan kotor disetiap barang yang laku terjual. Sistem simulasi penjualan dirancang untuk memprediksi barang dan mencari pendapatan kotor disetiap jenis barang pada *home-industry* Istiqomah berbasis web yang dibuat dengan menggunakan *PHP* dan *MySQL* sebagai database. Dengan sistem ini diharapkan dapat digunakan oleh *home-industry* untuk mensimulasikan penjualan barang pada toko, dan dapat mempermudah bagi pemilik *home-industry* dalam memperoleh informasi tentang kemungkinan yang akan terjadi terhadap penjualan produk keripik maco.

Kata Kunci : Produk, Monte Carlo, *PHP & MySQL*

KATA PENGANTAR

Puji syukur penulis sampaikan kehadirat Allah SWT yang telah memberikan syafat dan, hidayah serta karunia-nya sehingga penulis dapat menyelesaikan TugasAkhir (Skripsi)dan dapat pula menyelesaikan laporan ini. Shalawat dan salam penulis kirimkan kepada junjungan alam, kekasih Allah SWT Nabi Muhammad SAW yang telah menjadi tolak ukur, pedoman, dan bimbingan bagi kehidupan manusia dari kebodohan dan kegelapan menuju cahaya terang dengan kebenaran ilmu, iman, dan amal seperti saat ini.Dalam menyelesaikan Penelitian ini kami mengucapkan terima kasih kepada:

1. **Bapak H.Herman Nawas**, selaku Ketua Yayasan Universitas Putra Indonesia YPTK Padang.
2. **Bapak Dr. Sarjon Defit, S.Kom, M.Sc**, selaku Rektor Universitas Putra Indonesia YPTK Padang
3. **Bapak Dr. Julius Santony, S.Kom, M.Kom**, selaku Dekan Fakultas Ilmu Komputer Universitas Putra Indonesia YPTK Padang

Akhir kata, semoga skripsi ini dapat memberikan konstribusi terhadap masyarakat dan khusunya terhadap Universitas Putra Indonesia YPTK Padang. Penulis mengucapkan terima kasih dan mudah-mudahan laporan skripsi ini berguna dan dapat menambah wawasan bagi para pembaca.

Padang,3 Juli 2019

# Penulis

**DAFTAR ISI**

* 1. [Latar Belakang 1](#_TOC_250046)
	2. Rumusan Masalah 2
	3. [Hipotesa 3](#_TOC_250045)

1.4. Batasan Masalah.........................................................................................3 ..

* 1. [Tujuan Penelitian 4](#_TOC_250044)
	2. [Manfaat Penelitian 4](#_TOC_250043)

[BAB II LANDASAN TEORI](#_TOC_250042)

* 1. [Rekayasa Perangkat Lunak 6](#_TOC_250041)
		1. Tujuan Reakayasa Perangkat Lunak 7
		2. [Proses Rekayasa Perangkat Lunak 7](#_TOC_250040)
		3. [Model Rekayasa Perangkat Lunak 8](#_TOC_250039)
		4. Model Waterfall 9
		5. Model Prototipe 11
		6. Model RAD 12
		7. Model Iteratif 14
		8. Model Spiral 16
	2. Unified Modelling Language ( UML ) 17
		1. [Sejarah UML 17](#_TOC_250038)
		2. Diagram UML 18
		3. [Usecase Diagram 20](#_TOC_250037)
		4. [Class Diagram 22](#_TOC_250036)
		5. [Activity Diagram 24](#_TOC_250035)
		6. Sequence Diagram 26
		7. [Collaboration Diagram 28](#_TOC_250034)
		8. [Deployment Diagram 29](#_TOC_250033)
		9. [State Machine Diagram 30](#_TOC_250032)
	3. [Tentang Pemodelan Dan Simulasi 31](#_TOC_250031)
		1. [Tujuan Pemodelan Dan Simulasi 32](#_TOC_250030)
		2. [Istilah Dalam Simulasi 33](#_TOC_250029)
		3. [Pengertian Model 34](#_TOC_250028)
		4. [Jenis-jenis Model 35](#_TOC_250027)
		5. [Definisi Simulasi 36](#_TOC_250026)
		6. [Metode Monte Carlo 38](#_TOC_250025)
		7. [Simulasi Monte Carlo 38](#_TOC_250024)
		8. [Teknik Monte Carlo 39](#_TOC_250023)
		9. [Random Number 41](#_TOC_250022)
	4. [Pemograman Web 44](#_TOC_250021)
		1. [PHP 44](#_TOC_250020)
		2. [Sejarah PHP 45](#_TOC_250019)
		3. [Kelebihan PHP 46](#_TOC_250018)
		4. [Kekurangan PHP 46](#_TOC_250017)
	5. [Konsep Database 47](#_TOC_250016)
		1. [Pengertian Database 47](#_TOC_250015)
		2. [MySQL 47](#_TOC_250014)

[BAB III METODOLOGI PENELITIAN](#_TOC_250013)

* 1. Ker

angka penelitian 49

* 1. Ta

hapan Penelitian 50

* + 1. [Penelitian Pendahuluan 50](#_TOC_250012)
		2. [Pengumpulan Data 50](#_TOC_250011)
			1. [Waktu Penelitian 50](#_TOC_250010)
			2. [Tempat Penelitian 51](#_TOC_250009)
			3. [Metode Penelitian 51](#_TOC_250008)
		3. [Analisa 53](#_TOC_250007)
			1. [Analisa Data 53](#_TOC_250006)
			2. [Analisa Proses 53](#_TOC_250005)
			3. [Analisa Sistem 53](#_TOC_250004)
		4. [Perancangan 54](#_TOC_250003)
			1. [Perancangan Model 54](#_TOC_250002)
		5. [Implementasi 56](#_TOC_250001)
		6. [Pengujian Sistem 57](#_TOC_250000)

# BAB 1 PENDAHULUAN

# Latar Belakang

Teknologi komputer saat ini sudah semakin berkembang penggunaannya. Komputer yang pada awalnya digunakan sebagai alat hitung. Seiring dengan perkembangan zaman, komputer banyak digunakan di berbagai bidang. Misalnya pada bidang kesehatan, ekonomi dan sebagainya. Salah satu pemanfaatan teknologi computer yaitu dapat digunakan untuk sistem pakar.

Secara umum, sistem pakar adalah system yang berusaha mengadopsi pengetahuan manusia ke komputer yang dirancang untuk memodelkan kemampuan menyelesaikan masalah seperti layaknya seorang pakar. Ia akan memberi daftar gejala-gejala sampai bisa mengidentifikasikan suatu obyek berdasarkan jawaban yang diterima.

Sistem pakar dikembangkan dalam berbagai bidang, termasuk dalam bidang medis. Saat ini kebutuhan manusia akan pelayanan medis yang lebih baik sangat mendesak, yang berarti dukungan instrumentasi dan informatika medis modern (telemedis) menjadi sangat dibutuhkan termasuk metode untuk membantu analisisnya sehingga dihasilkan diagnosis yang lebih optimal. Contoh sistem pakar dalam bidang medis yaitu Sistem Pakar Untuk Diagnosa Penyakit Anak.

Infeksi Saluran Kemih (ISK) atau Urinarius Tractus Infection (UTI) adalah suatu istilah umum yang dipakai untuk mengatakan adanya infeksi pada saluran kemih. Dibandingkan laki-laki, perempuan mengidap ISK 10 kali lebih sering

dibandingkan laki-laki, karena jarak antara kandung kemih dengan kulit yang dipenuhi bakteri 5 cm pada perempuan dibandingkan dengan 20 cm pada laki-laki.

Diagnosis ISK dapat melalui pemeriksaan sampel urin dan pemeriksaan darah. Umumnya banyak orang enggan atau sungkan ke dokter ketika merasakan nyeri saat berkemih, anyang-anyangan (rasa ingin berkemih kembali usai berkemih), atau nyeri perut pada bagian bawah padahal berbagai keluhan tersebut sangatlah berarti sebagai alarm tubuh ketika terdapat suatu masalah pada sistem perkemihan atau sistem urinarial. Komplikasi potensial dari penyakit ISK mencakup gagal ginjal dan sepsis (infeksi yang meradang dan menyebar melalui aliran darah).

Salah satu cara untuk mendeteksi penyakit ISK secara dini dapat juga dilakukan dengan memanfaatkan teknologi canggih dan modern seperti yang diketahui dewasa ini, yaitu telah berkembangnya bidang studi Artificial Intelligence(AI) yang mempelajari serta mampu meniru kecerdasan manusia. Salah satu bagian dari kecerdasan buatan adalah sistem pakar (Expert system).

Dalam Tugas Akhir ini akan dibuat suatu sistem pakar yang nantinya akan menggunakan pendekatan dengan metode backward chaining adalah strategi untuk memprediksi atau mencari solusi dari suatu masalah yang dimulai dengan sekumpulan fakta yang diketahui yang kemudian akan menurunkan fakta baru (kesimpulan yang optimal).

Berdasarkan uraian diatas maka dalam penyusunan tugas akhir ini, penulis memilih judul “**PERANCANGAN APLIKASI SISTEM PAKAR UNTUK DIAGNOSIS PENYAKIT INFEKSI SALURAN KEMIH DENGAN METODE FORWARD CHAINING BERBASIS WEB”**

# Perumusan Masalah

Perumusan masalah dalam penelitian ini adalah :

* + 1. Bagaimana merancang dan membangun sistem pakar yang mampu mendiagnosa penyakit ISK secara valid sehingga pasien secara mudah dan cepat mendapatkan hasil diagnosa penyakit beserta keterangan dan solusi.
		2. Bagaimana mengimplementasikan forward chaining ke dalam sistem pakar sehingga menghasilkan perhitungan valid yang cepat dan akurat.
		3. Bagaimana aplikasi sistem pakar ini dapat digunakan dan diterapkan di kehidupan orang banyak dan membantu kehidupan masyarakat.

# Hipotesa

Dari perumusan masalah yang telah dikemukakan diatas, maka dapat di bentuk hipotesa sebagai berikut :

1. Sistem pakar dapat mengenali penyakit ISK, yang suatu waktu menyerang masyarakat tanpa batas usia dengan data yang di dapat dari seorang pakar.
2. Sistem pakar dapat memutuskan apakah seseorang terserang gejala penyakit ISK.
3. Sistem pakar dapat memberikan solusi terhadap permasalahan yang di hadapi penderita.
4. Diharapkan sistem pakar dengan metode forward chaining dapat membantu dalam menyelesaikan suatu masalah dengan menarik suatu kesimpulan.

# Batasan Masalah

Dalam perancangan dan pembuatan sistem pakar ini batasan-batasan atau ruang lingkupnya adalah :

* + 1. Menggunakan metode inferensi backward chaining untuk penarikan kesimpulan.
		2. Pembuatan sistem pakar ini hanya berdasarkan gejala-gejala umum dari penyakit ISK saja.
		3. Sistem pakar yang dirancang berfungsi untuk diagnosa dini dan penanganan awal penyakit ISK.
		4. Sistem pakar yang dirancang dapat digunakan langsung oleh user tanpa proses login dan penyimpanan data user tidak tersedia.

# Tujuan Penelitian

Adapun tujuan penelitian ini adalah :

* + 1. Memberikan kemudahan pada masyarakat dalam mendapatkan informasi tentang penyakit infeksi saluran kemih (ISK) secara cepat.
		2. Memberikan informasi mengenai penyebab, gejala-gejala ISK dan cara menanganinya.

# Manfaat Penelitian

Manfaat dari Penulisan Penelitian ini adalah :

* + 1. Membantu masyarakat agar dapat lebih mengetahui seluk-beluk penyakit ISK sehingga akan membantu proses komunikasi dengan pihak dokter ketika dibutuhkan penanganan lebih lanjut oleh tenaga medis yang sebenarnya.
		2. Aplikasi sistem pakar ini ditujukan kepada masyarakat umum untuk memberikan kemudahan mendeteksi penyakit ISK secara dini serta penanganan awalnya.
		3. Pakar dapat memberikan suatu informasi kepada masyarakat melalui tanya jawab dengan sistem komputerisasi.

# BAB II LANDASAN TEORI

# Rekayasa Perangkat Lunak

Rekayasa perangkat lunak (RPL) lebih fokus pada praktik pengembangan pelanggan (*customer*). Adapun ilmu komputer lebih fokus pada teori dan konsep dasar perangkat komputer. Pearangkat lunak yang baik adalah perangkat yang fokus pada pengguna atau pelanggan (Rosa & M.Shalahudin, 2014). Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut :

* + 1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*).
		2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependability dan robust*).
		3. Efisien dari segi sumber daya dan pengguna.
		4. Kemampuan untuk dipakai sesuai dengan kebutuhan (*usability*).

Dari kriteria diatas maka perangkat lunak yang baik adalah perangkat lunak yang dapat memenuhi kebutuhan pelanggan (*customer*) atau pemakai perangkat lunak (*user*) atau berorientasi pada pelanggan atau pemakai perangkat lunak, bukan berorientasi pada pembuat atau pengembang perangkat lunak.

# Tujuan Rekayasa Perangkat Lunak

Bidang rekayasa akan selalu berusaha menghasilkan *output* yang kinerjanya tinggi, biaya rendah dan waktu penyelesaian yang tepat. Secara lebih khusus kita dapat menyatakan tujuan RPL adalah :

* + - 1. Memperoleh biaya produksi perangkat lunak yang rendah.
			2. Menghasilkan perangkat lunak yang kinerjanya tinggi, handal dan tepat waktu.
			3. Menghasilkan perangkat lunak yang dapat bekerja pada berbagai jenis

*platform*.

* + - 1. Menghasilkan perangkat lunak yang biaya perawatannya rendah.

# Proses Rekayasa Perangkat Lunak

Proses rekayasa perangkat lunak dilakukan selama pembangunan perangkat lunak. Proses-proses yang dilakukan dalam rekayasa perangkat lunak secara garis besar adalah sebagai berikut pada gambar 2.1 :

Mulai

Pengujian

Analisis Implementasi

Perancangan

***Sumber : Rosa A.S M.Salahuddin (2014)***

# Gambar 2.1 Tahapan Umum Rekayasa Perangkat Lunak

Proses-proses pada gambar di atas dapat dilakukan berulang kali sampai perangkat lunak memenuhi kebutuhan pelanggan atau user. Untuk membangun perangkat lunak yang benar-benar baik maka diperlukan tahapan-tahapan rekayasa perangkat lunak. Perangkat lunak yang dibangun tanpa melalui tahapan rekayasa perangkat lunak yang baik maka akan seperti membuat baju tanpa pola dan rencana model baju yang baik. Dalam membuat sebuah baju, maka dibutuhkan adanya perencanaan model baju dan pembuatan pola baju yang baik. Setelah pola baju terbentuk maka kain mulai dipotong sesuai pola, baru kemudian dijahit. Jika baju dibuat tanpa pola, maka yang terjadi adalah baju hasil tambal sulam.

Seperti halnya pembuatan perangkat lunak tanpa melalui proses rekayasa perangkat lunak, maka hanya akan menghasilkan perangkat lunak “tambal sulam”, begitu kurang maka akan ditambal. Tentu saja hasil perangkat lunak “tambal sulam” tidak akan bagus. Rekayasa perangkat lunak bukan seperti sulap yang bisa diselesaikan dalam sekejap, ada beberapa proses yang harus di lalui agar menghasilkan perangkat lunak yang berkualitas (Rosa A.S dan M. Shalahuddin, 2014).

# Model Rekayasa Perangkat Lunak

Pada rekayasa perangkat lunak, banyak model yang telah dikembangkan untuk membantu proses pengembangan perangkat lunak. Model-model ini pada umumnya mengacu pada model pengembangan sistem yang disebut *System Development Live Cycle* (SDLC).

SDLC atau Software Development Life Cycle atau sering disebut juga System Development Life Cycle adalah proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (Rosa A.S dan M. Shalahuddin, 2014).

SDLC memiliki model-model dalam penerapan tahapan prosesnya. Model- model tersebut diantaranya:

* + 1. **Model *Waterfall***

Model SDLC air terjun (*waterfall*) sering juga disebut model sekuensial linier (*sequential linear*) atau alur hidup klasik (*classic life cycle*). Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis, desain, pengodean, pengujian, dan tahap pendukung (*support*). Berikut adalah gambar 2.2 model air terjun:



***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Gambar 2.2 Ilustrasi Model Waterfall

1. Analisis kebutuhan perangkat lunak

Proses pengumpulan kebutuhan dilakukan secara intensif demi mespesifikasikan kebutuhan perangkat lunak supaya dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*.

1. Desain

Desain perangkat lunak adalah proses multi langkah yang terfokus pada desain perancangan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka, dan prosedur coding.

1. Pembuatan kode program

Desain harus di translasikan kedalam program perangkat lunak, hasil dari tahap ini adalah program komputer yang harus sesuai dengan desain yang telah dibuat pada tahap sebelumnya.

1. Pengujian

Pengujian fokus pada perangkat lunak dari segi logis dan fungsional serta memastikan bahwa semua bagian sudah diuji.

1. Pendukung dan pemeliharaan

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke user. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru.

Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifiaksi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru (A.S & M.Shalahudin, 2014).

# Model Prototype

Model Prototype merupakan suatu paradigma baru dalam metode pengembangan perangkat lunak dimana metode ini tidak hanya sekedar evolusi dalam dunia pengembangan perangkat lunak, tetapi juga merevolusi metode pengembangan perangkat lunak yang lama yaitu sistem sekuensial yang biasa dikenal dengan nama SDLC atau Waterfall Development Model di jelaskan pada gambar 2.3.

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Gambar 2.3 Ilustrasi Model Prototipe

*Mock-up* adalaah sesuatu yang digunakan sebagai model desain yang digunakan untuk mengajar, demonstrasi, evaluasi desain, promosi, atau keperluan

lain. Sebuah *Mock-up* disebut sebagai prototipe perangkat lunak jika menyediakan atau mampu mendemonstrasikan sebagian besar fungsi sistem perangkat lunak dan memungkinkan pengujian desain sistem perangkat lunak. Iterasi terjadi pada pembuatan prototipe sampai sesuai dengan keinginan pelanggan *(costumer)* atau *user.* Model prototipe cocok digunakan untuk menggali spesifikasi kebutuhan pelanggan secara lebih detail tetapi beresiko tinggi terhadap membengkaknya

biaya dan waktu proyek (Rosa & M.Shalahudin, 2014).

* + 1. **Model *Rapid Application Development* (RAD)**

*Rapid Application Development* (RAD) adalah model proses pengembangan perangkat lunak yang bersifat inkremental terutama untuk waktu pengerjaan yang pendek. Model RAD adalah adaptasi dari model air terjun versi kecepatan tinggi dengan menggunakan model air terjun untuk pengembangan setiap komponen perangkat lunak (Rosa & M.Shalahudin, 2014).

Metode RAD merupakan pendekatan berorientasi objek untuk menghasilkan sebuah sistem dengan sasaran utama mempersingkat waktu pengerjaan aplikasi dan proses agar sesegera mungkin memberdayakan sistem perangkat lunak tersebut secara tepat dan cepat. Untuk perancangan suatu sistem informasi yang normal seandainya membutuhkan waktu minimal 180 hari, maka dengan menerapkan metode RAD hanya membutuhkan waktu 30-90 hari untuk menyelesaikan sistem perangkat lunak tersebut. dapat dijelaskan pada gambar 2.4.



***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Gambar 2.4 Metode RAD (Rapid Application Development)

1. Pemodelan Bisnis

Pemodelan yang dilakukan untuk memodelkan fungsi bisnis untuk mengetahui informasi apa yang terkait proses bisnis, informasi apa saja yang harus dibuat, siapa yang harus membuat informasi itu, bagaimana alur informasi itu, proses apa saja yang terkait informasi itu.

1. Pemodelan Data

Memodelkan data apa saka yang dibutuhkan berdasarkan pemodelan bisnis dan mendefinisikan atribut-atributnya beserta relasi dengan data-data yang lain.

1. Pemodelan Proses

Mengimplementasikan fungsi bisnis yang sudah didefinisikan terkait dengan pendefinisian data.

1. Pembuatan Aplikasi

Mengimplementasikan pemodelan proses dan data menjadi program. Model RAD sangat menganjurkan pemakaian komponen yang sudah ada jika dimungkinkan.

1. Pengujian dan Pergantian

Menguji komponen-komponen yang dibuat. Jika sudah teruji maka tim pengembang komponen dapat beranjak untuk mengembangkan komponen berikutnya.

* + 1. **Model *Iteratif***

Model iteratif merupakan model yang diciptakan untuk mengatasi beberapa kekurangan model *waterfall*. Model ini bisa dikatakan sebagai mini *waterfall*. Di manasetiap subbagian sekuensial dari pengembangan program, di dalamnya disusun dengan mengikuti fase analisis, desain, implementasi, dan pengujian. Hasil pengujian dipakai sebagai dasar dalam mengerjakan subbagian selanjutnya. Model iteratif merupakan gabungan dari model *waterfal* dan model prototipe, model ini cocok digunakan pengembang dengan *turnover* staf yang tinggi (Rosa & M.Shalahudin, 2014). Berikut adalah gambar 2.5 dari model inkremental:



***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

**Gambar 2.5 Ilustrasi Model *Iteratif***

Model inkremental dibuat untuk mengatasi kelemahan dari model air terjun yang tidak mengakomodasikan iterasi, dan mengatasi kelemahan dari metode prototipe yang memiliki proses terlalu pendek dan setiap iteratif prosesnya tidak selalu menghasilkan produk.

* + 1. **Model *Spiral***

Model *spiral* (*spiral model*) memasangkan iteratif pada model prototipe dengan kontrol dan aspek sistematik yang diambil dari model air terjun. Model spiral menyediakan pengembangan dengan cara cepat dengan perangkat lunak yang memiliki versi yang terus bertambah fungsinya *(increment).* Model spiral cocok digunakan untuk mengembangkan aplikasi dengan skala besar tetapi target waktu dan biaya tidak terlalu meningkat (Rosa & M.Shalahudin, 2014). Berikut adalah gambar 2.6 dari model spiral:



***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Gambar 2.6 Ilustrasi Model *Spiral*

Model spiral dilakukan searah jarum jam dimulai dari sumbu proyek. Sumbu proyek dapat digunakan sebagai awal iterasi ataupun evaluasi dari iterasi yang sudah dilakukan. Pada gambar diatas setiap wilayah berputar dengan urutan kerja tertentu.

# Unified Modeling Language (UML)

Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung.

Unified Modeling Language(UML) merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teksteks pendukung (Andrianof, 2017).

UML hanya berfungsi untuk melakukan pemodelan. UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak

digunakan pada metodologi berorientasi objek.

# Sejarah UML

Bahasa pemrograman berorientasi objek yang pertama dikembangkan dikenal dengan nama Simula-67 yang dikembangkan pada tahun 1967. Bahasa pemrograman

ini kurang berkembang dan dikembangkan lebih lanjut, namun dengan kemunculannya telah memberikan sumbangan yang besar pada developer pengembang bahasa pemrograman berorientasi objek selanjutnya.

Karena banyaknya metodologi-metodologi yang berkembang pesat saat itu, maka muncullah ide untuk membuat sebuah bahasa yang dapat dimengerti semua orang. Usaha penyatuan ini banyak mengambil dari metodologi-metodologi yang berkembang saat itu. Maka dibuat bahasa yang merupakan gabungan dari beberapa konsep seperti konsep *Object Modelling Technique (OMT)* dari Rumbaugh dan Booch (1991), konsep *The Classes, Responsibilities, Collaboratos* (CRC) dari Rebecca Wirfs-Brock (1990), konsep pemikiran Ivar Jacobson, dan beberapa konsep lainnya dimana James R. Rumbaigh, Grady Booch, dan Ivar Jacobson bergabung dalam sebuah perusahaan yang bernama *Rational Software Corporation* menghasilkan bahasa yang disebut dengan *Unified Modelling*

*Language* (UML) (Rosa & M.Shalahudin, 2014) .

* + 1. **Diagram *Unified Modelling Language (UML)***

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.7 dibawah.

UML 2.3

Structure

Behaviour

Interaction Diagram

Class Diagram Object Diagram

Component Diagram

Composite Strutcture Diagram

Package Diagram

Use Case Diagram

Activity Diagram

State Machine Diagram

Sequence Diagram

Communication Diagram

Timing Diagram

Interaction Overview Diagram

Deployment Diagram

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Gambar 2.7 Pembagian Kategori

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

* + - 1. *Structure diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
			2. *Behavoiur diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
			3. *Interaction diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.
		1. ***Use Case Diagram***

*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behaviour*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Menggambarkan sekelompok Use cases dan aktor yang disertai dengan hubungan diantaranya. Diagram Use case ini menjelaskan dan menerangkan kebutuhan atau requirement yang diinginkan user,(Andrianof, 2017). Use case atau diagram use case merupakan pemodelan untuk kelakuan (behavior) sistem informasi yang akan dibuat(Saiful & Ambarita, 2017).

Diagram aktivitas atau activitydiagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak yang di jelaskan pada tabel 2.1 dibawah.

**Tabel 2.1 : *Use case diagram***

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Use CaseNewUseCase | Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case* |



|  |  |
| --- | --- |
| Aktor / *actor*Aktor | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol yang dibuat buat aktor belum tentu merupakan orang; biasanya dinyatakan dengan menggunakan kata benda di awal frase nama aktor. |
| Asosiasi / association asosiasi  | Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan actor |
| Ekstensi / extendExtensi / extend | Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip *inherintance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan use case yang ditambahakan |
| Generalisasi / *generalization*Generalization | Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya. |
| Menggunakan / *include* / *uses*<<include>> <<uses>>  | Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

Setiap *use case* dilengkapi dengan skenario. Skenario *use case* adalah alur jalannya proses *use case* dari sisi aktor dan sistem. Berikut adalah tabel format tabel

2.2 di bawah ini*.*

**Tabel 2.2 : Skenario *use case***

|  |  |
| --- | --- |
| **Aksi Aktor** | **Reaksi Sistem** |
| Skenario Normal |
|  |  |
| Skenario Alternatif |
|  |  |

Skenario *use case* dibuat per *use case* terkecil, misalkan untuk generalisasi maka skenario yang dibuat adalah *use case* yang lebih khusus. Skenario normal adalah skenario yang berjalan dengan normal tanpa terjadi kesalahan atau *error.* Sedangkan skenario alternatif adalah skenario bila sistem tidak berjalan normal, atau mengalami *error.* Skenario normal dan skenario alternatif dapat lebiih dari satu. Alur dari skenario inilah yang nantinya menjadi dasar pembuatan diagram

sequen.

* + 1. ***Class Diagram***

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. Class menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (Andrianof, 2017).

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 metode atau operasi.

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

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas sesuai dengan rancangan di dalam diagram kelas agar antara dokumentasi dan perangkat lunak sinkron. Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai kebutuhan sistem, sehingga pembuat perangkat lunak atau *programmer* dapat memebuat kelas-kelas didalam program perangkat lunak sesuai dengan perancangan diagram kelas. Berikut simbol-simbol yang ada pada tabel 2.3 di bawah ini.

**Tabel 2.3 : *Class diagram***

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Kelas | Kelas pada struktur system |
| Antarmuka / *interface* | Sama dengan konsep *interface* dalam pemrograman berorientasi objek |



|  |  |
| --- | --- |
| Interface |  |
| Asosiasi / *association* asosiasi  | Relasi antar kelas dengan makna umum, asosiasi biasanya disertai dengan *multiplicity* |
| Asosiasi berarah / *directed association* | Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga dengan disertai *multiplicity* |
| GeneralisasiGeneralization | Relasi antar kelas dengan makna generalisasi spesialisasi (umum- khusus) |
| Kebergantungan / *dependency* | Relasi antar kelas dengan makna kebergantungan antar kelas |
| Agregasi / aggregation | Relasi antar kelas dengan makna semua bagian (*whole-part*) |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

* + 1. ***Activity Diagram***

Activity Diagram menggambarkan workflow (aliran Kerja) atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktifitas mengambarkan aktifitas sistem bukan apa yang dilakukan aktor, jadi aktifitas yang dapat dilakukan oleh sistem (Rosa & M.Shalahudin, 2014).

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis dimana setiap uruan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokkan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.
3. Rancangan pengujian dimana setiap akktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya.
4. Rancangan menu yang ditampilkan diperangkat lunak.

**Tabel 2.4 : *Activity diagram***

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| Status awal | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
| AktivitasNewActivity | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| Percabangan / *decision* | Asosiasi percabangan dimana lebih dari satu aktivitas digabungkan menjadi satu |
| Penggabungan / *join* | Asosiasi penggabungan dimana lebiih dari satu aktivitas digabungkan menjadi satu |
| Status akhir | Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir |
| Swimlanesimbol swimlane.pngAtau | Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi |

|  |  |
| --- | --- |
| simbol swimlane miring.png |  |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

* + 1. **Sequence *Diagram***

Sequence diagram menggambarkan kelakuan objek pada use case dengan mendeskrisikan waktu hidup objek dan message yang dikirim dan diterima antar objek (Andrianof, 2017).

Diagram sekuen menggambarkan kelakuan objek pada *use case* dan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat sekuen diagram 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 pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semankin banyak. Berikut adalah simbol-simbol pada tabel

* 1. di bawah ini:

**Tabel 2.5: *Sequence diagram***

|  |  |
| --- | --- |
| Simbol | Deskripsi |
| AktorAktorAtaunama aktorTanpa waktu aktif Garis hidup / *lifeline* | Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walauppun simbol dari aktor adalah orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda diawal frase nama aktor |
| Objek | Menyatakan objek yang berinteraksi |



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | nama objek : nama kelas |  |  | pesan |
|  |
| Waktu aktif | Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya. |
| Pesan tipe create | Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat |
| Pesan tipe call | Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri |
| Pesan tipe send | Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah kepada objek yang dikirimi |
| Pesan tipe return keluaran  | Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah kepada objek yang menerima kembalian |
| Pesan tipe destroy | Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah padda objek yang diakhiri, sebaiknya jika ada create maka ada destroy |
|  |  |
|  |  |  |
|  |  |
|  |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

* + 1. ***Collaboration Diagram***

Diagram komunikasi menggambarkan interaksi antar objek/bagian dalam bentuk urutan pengiriman pesan. Berikut adalah simbol-simbol yang ada pada tabel

* 1. di bawah ini.

**Tabel 2.6 Simbol pada *Communications Diagram***

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |
| Objeknama\_objek : nama\_kelas | Objek yang melakukan interaksi pesan |
| *Link* | Relasi antar-objek yang menghubungkan objek satu dengan yang lainnya atau dengan dirinya sendirinama\_objek : nama\_kelas |
| Arah pesan / *stimulus* | Arah pesan yang terjadi, jika pada suatu *link* ada dua arah pesan yang berbeda maka arah juga digambarkan dua arah pada dua sisi *link*. |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

* + 1. ***Deployment Diagram***

*Deployment* diagram memberikan gambaran dari arsitektur fisik perangkat lunak, perangkat keras, dan artefak dari sistem. Berikut simbol-simbol yang ada pada tabel 2.7 diagram *deployment*:

**Tabel 2.7 Simbol pada *Deployment Diagram***

|  |  |
| --- | --- |
| **Simbol** | **Deskripsi** |

|  |  |
| --- | --- |
| *Package*package | *Package* merupakan sebuah bungkusan dari satu atau lebih *node*. |
| *Node* | Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak yang tidak dibuat sendiri (*software*), jika di dalam *node* disertakan komponen untuk mengkonsistenkan rancangan maka komponen yang diikutsertakan harus sesuai dengan komponen yang telah di definisikansebelumnya pada diagram komponen. |
| Kebergantungan/*dependency* | Ketergantungan antar *node*, arah panah mengarah pada *node* yang dipakai. |
| *Link* | Relasi antar *node*. |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

* + 1. ***State Machine Diagram***

Berikut ini komponen-komponen dasar yang ada dalam *state machine diagram*

di jelaskan pada tabel 2.8 :

**Tabel 2.8 Simbol pada *State Machine Diagram***

|  |  |
| --- | --- |
| **SIMBOL** | **KETERANGAN** |
| *Start/*Status Awal*(Initial State)*  | Keadaan awal pada saat system mulai hidup. |



|  |  |
| --- | --- |
| *End /* Status Akhir (*Final State)* | Keadaan akhir dari daur hidup suatu system. |
| *Event* | Kegiatan yang menyebabkan berubahnya status mesin. |
| *State* | Sistem pada waktu tertentu. *State* dapat berubah jika ada *event* tertentu yang memicu perubahan tersebut. |

***Sumber : Rosa A.S dan M. Shalahuddin (2014)***

# Tentang Pemodelan Dan Simulasi

Pemodelan dan simulasi didasari oleh pengetahuan komputer, matematika serta probabilitas dan statistik, walaupun tak dapat disangkal bahwa prosesnya sering masih dilakukan sebagai proses yang instingtif. Karena untuk membantu suatu model dari sistem dan melakukan simulasi tidak ada rumus-rumus yang eksak, maka yang dapat diberikan hanyalah petunjuk-petunjuk secara garis besar dalam bidang tersebut. Suatu model dan simulasi yang spesifik harus dilakukan terhadap suatu sistem yang bersangkutan, tetapi suatu analogi dapat diambil terhadap model yang telah ada sebagai acuan.

# Tujuan Pemodelan Dan Simulasi

Dalam pandangan sistem, pemodelan dan simulasi dapat digunakan untuk tujuan berikut:

* + - 1. Studi perilaku sistem komplek, yaitu sistem dimana suatu solusi analitik tidak dapat dilakukan.
			2. Membandingkan alternatif rancangan untuk suatu sistem yang tidak atau belum ada.
			3. Studi pengaruh perubahan terhadap sistem yang ada dengan tanpa merubah sistem.
			4. Memperkuat atau memverifikasi suatu solusi analitik.

Beberapa keuntungan simulasi sebagai berikut :

1. Simulasi mengizinkan keluwesan besar dalam pemodelan sistem komplek, sehingga model simulasi dapat sangat *valid.*
2. Mudah membandingkan berbagai alternatif.
3. Kendali kondisi eksperimental.
4. Dapat mempelajari sistem dengan bingkai waktu yang sangat panjang.

Simulasi umumnya menghasilkan volume besar keluaran sehingga perlu untuk meringkaskan, sesuai analisis statistic.

# Istilah Dalam Simulasi

Pengertian sistem adalah sehimpunan unsur yang melakukan sesuatu kegiatan atau menyusun skema atau tata cara melakukan sesuatu kegiatan pengolahan (pemrosesan) untuk mencapai sesuatu atau beberapa tujuan, dan hal ini dilakukan dengan cara mengolah data dan/atau energi dan/atau barang (benda) di dalam jangka waktu tertentu guna menghasilkan informasi dan/atau energi dan/atau barang (benda).

Kejadian (*event*) adalah peristiwa sesaat yang dapat mengubah variabel keadaan sistem. Biasanya kejadian terjadi dengan selang-selang waktu acak. Suatu kejadian dapat mengubah harga dari atribut, mengubah entitas, atau memulai/mengakhiri suatu aktifitas.

Model adalah representasi dari dari suatu obyek, benda atau ide-ide dalam bentuk yang lain dengan entitasnya. Model berisi informasi tentang suatu sistem yang dibuat dengan tujuan untuk mempelajari perilaku sistem yang sebenarnya.

Variabel sistem adalah besaran yang nilainya berubah setiap saat baik secara kontinu maupun diskret dalam sistem. Elemen dari suatu sistem diwakili oleh atributnya, dan pengertian tentang variabel dalam pemodelan adalah suatu atribut tertentu yang bernilai tidak tetap, baik atribut elemen sistem, maupun atribut lingkungan sistem.

Keadaan (*state*) suatu sistem dinamis adalah himpunan minimal variabel- variabel sedemikian rupa sehingga pengetahuan tentang variabel tersebut pada waktu tertentu bersamaan dengan pengetahuan tentang masukan (*input*) secara lengkapa akan menentukan perilaku (*behaviour*) sistem pada waktu tersebut. Jadi keadaan (*state*) suatu sistem adalah kumpulan variabel-variabel dan nilainya diperlukan untuk mensifatkan suatu sistem pada waktu tertentu.

Parameter sistem adalah besaran yang nilainya akan memberi sifat (karakteristik) tertentu dari suatu sistem. Nilai parameter dapat tetap (*time invariant*) berupa konstanta atau berubah dengan waktu (*time variant*).

Simulasi adalah proses implementasi model menjadi program komputer (*software*) atau rangkaian elektronik dan mengeksekusi *software* tersebut sedemikian rupa sehingga perilakunya menirukan atau menyerupai sistem nyata (realitas) tertentu untuk tujuan mempelajari perilaku (*behaviour)* sistem, pelatihan ( *training* ) atau permainan ( *gamin* ) yang melibatkan sistem nyata (realitas).

# Pengertian Model

Model adalah representasi dari suatu objek, benda atau ide-ide dalam bentuk yang lain dengan entitasnya.

Tujuan dari studi pemodelan adalah menentukan informasi (variabel dan parameter) yang dianggap penting untuk dikumpulkan, sehingga tidak ada model yang unik. Satu sistem dapat mempunyai berbagai model, tergantung dari sudut pandang dan kepentingan si pembuat model. Sebaliknya, model yang sama dapat dipakai untuk berbagai sistem, jika perilaku dari sistem-sistem tersebut sama.

# Jenis-Jenis Model

Model dapat direpresentasikan dalam berbagai cara, tergantung tujuan dari studi. Beberapa jenis model dapat dikelompokkan sebagai berikut :

1. Model Skala : adalah model yang dibuat biasanya dengan memeperkecil skala dari aslinya. Misalnya : model mobil, model pesawat terbang untuk uji terowongan angin (*wind tunnel*).
2. Model Piktorial (Visual Grafis) : model yang dibuat dengan menggambar rancangan yang sebenarnya belum ada. Misalnya : designer menggambar model baju, arsitek menggambar rumah.
3. Model Verbal : model yang penjelasannya dengan kata-kata. Misalnya: proses inflasi tergantung dari beberapa faktor ekonomi makro, dijelaskan dengan kata-kata baru dibuat diagram skematis.
4. Model Skematis : model yang melukiskan unsur-unsur sistem dalam bentuk skema, petak-petak dan arus barang atau informasi. Model diagram dapat berupa diagram, seperti: diagram blok, DFD (*Data Flow Diagram*), CD (*Context Diagram*), Petri Net, Flowchart, PERT (*Program Evaluation and Review Technique*). Model dalam bentuk skema statik, seperti: tabel, bagan Gantt (*Gantt Chart)*.
5. Model Simbolik (Matematika) : model dalam bentuk persamaan matematika, seperti: persamaan diferensial, persamaan diferens, persamaan aljabar, persamaan logika, dan lain-lain.
6. Model Komputer : model dalam bentuk program komputer (*source code*) yang ditulis menggunakan bahasa komputer tertentu, seperti: C, Pascal, Ada, dan lain-lain.

# Definisi Simulasi

Simulasi adalah suatu metode yang penting karena keunggulan dalam memperbaiki kinerja suatu sistem tanpa mengganggu kelancaran kerja sistem nyata (real system). Menurut (Riyanto, 2017) Simulasi adalah suatu metode yang penting karena keunggulan dalam memperbaiki kinerja suatu sistem tanpa mengganggu kelancaran kerja sistem nyata (real system). Sebuah model simulasi dikembangkan untuk mempelajari kerja sistem yang berkembang dari waktu ke waktu. Teknik simulasi diskrit adalah salah satu model simulasi pengoperasian sistem sebagai urutan peristiwa diskrit. Simulasi adalah proses implementasi model menjadi program komputer (*software*) atau rangkaian elektronik dan mengeksekusi *software* tersebut sedemikian rupa sehingga perilakunya menirukan atau menyerupai sistem nyata (realitas) tertentu untuk tujuan mempelajari perilaku (*behaviour*) sistem, pelatihan (*training*), atau permainan (*gaming*). Jadi, simulasi adalah proses merancang model dari suatu sistem yang sebenarnya, mengadakan percobaan- percobaan terhadap model tersebut dan mengevaluasi hasil percobaan tersebut.

Simulasi merupakan suatu metode eksperimental dan terpakai untuk: menjelaskan perilaku sistem, membangun teori atau hipotesis yang mempertanggungjawabkan perilaku dari sistem yang diamati, memakai teori-teori untuk meramalkan perilaku sistem yang akan datang, yaitu pengaruh yang akan

dihasilkan oleh perubahan-perubahan variabel dan parameter sistem atau perubahan operasinya.

Berdasarkan teknik atau metodologi simulasi, maka ada beberapa jenis simulasi, antara lain :

* + - 1. Simulasi *Monte Carlo,* tidak elemen waktu, digunakan untuk mengevaluasi ekspresi non-probabilistik (misalnya integral) menggunakan metode probabilistik, melibatkan banyak persoalan matematika.
			2. Simulasi kemudi-jejak, secara ekstensif digunakan misalnya pada evaluasi unjuk kerja sistem komputer, misalnya algoritma penghalaman (*paging*).
			3. Simulasi kejadian diskret, adalah suatu simulasi yang menggunakan model sistem kejadian diskret (*discrete event*), misalnya pada studi unjuk kerja sistem komputer digital, studi sistem antrian bank, dan lain-lain.
			4. Simulasi dinamis kontinu, menggunakan model keadaan perubahan kontinu terhadap waktu, misalnya pada studi proses reaksi kimia, gerakan dinamis suatu kendaraan (*vehicle*) baik darat, laut ataupun darat.

# Metode Monte Carlo

Metode Monte Carlo merupakan metode analisis numerik yang melibatkan pengambilan sampel eksperimen bilangan acak. Salah satu model simulasi yang paling popular pada pengendalian persediaan adalah simulasi Monte Carlo. Model simulasi Monte Carlo merupakan bentuk simulasi probabilistik dimana solusi dari suatu masalah diberikan berdasarkan proses Randomisasi (acak). Proses acak ini melibatkan suatu distribusi probabilitas dari variabel-variabel data yang dikumpulkan berdasarkan data masa lalu maupun distribusi probabilitas teoritis. Bilangan acak digunakan untuk menjelaskan kejadian acak dan secara berurutan mengikuti perubahan-perubahan yang terjadi dalam proses simulasi (Putra & Hadi, 2016).

Karena algoritma ini memerlukan pengulangan (repitisi) dan perhitungan yang amat kompleks, metode Monte Carlo pada umumnya dilakukan menggunakan komputer, dan memakai berbagai teknik simulasi komputer. Algoritma Monte Carlo adalah metode Monte Carlo numerik yang digunakan untuk menemukan solusi

problem matematis (yang dapat terdiri dari banyak variabel) yang sudah dipecahkan, misalnya dengan kalkulus integral, atau metode numerik lainnya.

# Simulasi Monte Carlo

Simulasi merupakan salah satu bentuk pemodelan terhadap sistem nyata untuk memudahkan mempelajari sistem kompleks. Fasilitas atau proses tersebut disebut dengan sistem, yang mana di dalam keilmuan digunakan untuk membuat asumsi-asumsi bagaimana sistem tersebut bekerja, dimana asumsi-asumsi tersebut biasanya berbentuk hubungan matematik atau logika yang akan membentuk model yang digunakan untuk mendapatkan pemahaman bagaimana perilaku hubungan dari sistem tersebut (Lambertus, Jairus, & KallaYoseph, 2015).

Simulasi Monte Carlo dikategorikan sebagai metode pencuplikan (*sampling*) sebab masukan dibangkitkan secara acak dari distribusi probabilitas untuk menyimulasikan proses pencuplikan dari populasi sebenarnya. Kelebihan lain dari simulasi Monte Carlo adalah waktu eksekusi program lebih baik disemua iterasi (Lambertus et al., 2015).

# Teknik Monte Carlo

Teknik Monte Carlo merupakan pendekatan khusus yang sangat berguna untuk mensimulasikan situasi yang mengandung risiko sehingga diperoleh jawaban- jawaban perkiraan yang tidak dapat diperoleh dari penelitian-penelitian secara fisik atau dari penggunaan analisis matematika. Proses Monte Carlo dalam memilih angka acak berdasarkan distribusi probabilitas bertujuan untuk menentukan variabel acak melalui uji sampel dari distribusi probabilitas. Teknik ini dapat dikerjakan dengan alat bantu yaitu perangkat lunak dari komputer beruupa lembar kerja (*Spreadsheet*) yang diprogram untuk membangkitkan bilangan random sesuai dengan yang dibutuhkan. Memunculkan nilai bilangan acak seperti : tabel bilangan acak, prosedur atau subrutin di dalam program.

Teknik simulasi Monte Carlo terbagi atas lima langkah sederhana yaitu sebagai berikut :

* + - 1. Menetapkan sebuah distribusi probabilitas bagi variabel penting. Ide dasar simulasi *Monte Carlo* adalah untuk membangkitkan nilai untuk variabel pada model yang sedang diuji. Dalam sistem dunia nyata, sebagian besar variabel memiliki probabilitas alami
			2. Membuat distribusi probabilitas kumulatif bagi setiap variabel. Untuk mengubah distribusi probailitas biasa menjadi sebuah distribusi kumulatif (*cumulative probability distribution*) merupakan pekerjaan yang mudah.
			3. Menetapkan sebuah interval angka acak bagi setiap variabel. Setelah distribusi probabilitas kumulatif bagi setiap variabel yang digunakan dalam simulasi sudah diterapkan, maka diberikan serangkaian angka yang mewakili setiap nilai atau output yang mungkin, angka ini disebut sebagai interval angka acak (*random-number interval*).
			4. Membangkitkan angka acak yang sedang diteliti melibatkan banyak percobaan simulasi, maka digunakan program komputer untuk membangkitkan angka acak. Jika simulasi dilakukan dengan perhitungan tangan, angka acaka dapat diambil dari sebuah tabel angka acak.
			5. Membangkitkan serangkaian percobaan. Hasil dari eksperimen dapat disimulasikan secara sederhana dengan memilih angka acak dari tabel angka acak.

# Random Number

Bilangan acak atau bilangan random adalah suatu bilangan yang tidak dapat diprediksi kemunculannya disebut dengan bilangan acak atau random. Ada beberapa algoritma pembangkit bilangan *random* atau *pseudo random number* yang digunakan seperti LCG (*Linier Congruential Generators*), MRNG (*Multiplicative Random Number Generator*). LCG merupakan metode yang digunakan untuk membangkitkan bilangan random dengan distribusi *uniform*. MRNG adalah metode pembangkitan bilangan random berupa bilangan-bilangan prima. Ketika membangkitkan *pseudo- random numbers* terdapat kesalahan tertentu seperti :

* + - 1. Angka yang dihasilkan tidak berdistribusi seragam.
			2. Angka yang dihasilkan adalah *discrete-valued* sebagai ganti nilai kontinyu.
			3. Rata-rata angka yang dihasilkan bernilai adalah terlalu tinggi atau terlalu rendah.
			4. Tidak independen, misalnya masih terdapat nilai auto korelasi antar angka- angka yang dihasilkan.

Suatu bilangan random harus *replicable.* Pembangkitan bilangan random memiliki beberapa metode, salah satunya yaitu *Conruential Method (LCM).* LCM menghasilkan suatu urutan bilangan bulat x1, x2,, antara 0 dan m-1 menurut hubungan berulang berikut ini :

*X1+1 = (a.Xi + C)* mod m, dengan 1 = 0,1,2… Dimana :

a : Konstanta perkalian

Xi : Nilai awal yang ditentukan C : Kenaikan

Mod : Modulus

Jika C ≠ 0, maka dikatakan sebagai *Mixed Congruential Method.* Ketika C = 0, maka dikatakan sebagai *Multiplicative Congruential Method.* Bilangan random antara 0 dan 1 dihasilkan oleh : Ri = Xi / m, I = 0,1,2..

Contoh Soal sebagai berikut, Carilah 3 bilangan acak mengunakan metode mixed congruent, dengan nilai awal Z0=12357, a=19, c=237, m=128. Gunakan tingkat ketelitian 4 digit di belakang koma.

Bilangan acak 1

Zi+1 = (a . Zi + c) mod m

= (19 . 12357 + 237) mod 128

= 235020 mod 128

= 12 (Random Number)

Random Variate :

R1 = Zi+1 / m

= 12 /128

= **0.0938**

Bilangan acak 2

Zi+1 = (a . Zi + c) mod m

= (19 . 12 + 237) mod 128

= 465 mod 128

= 81 (Random Number) Random Variate :

R2 = Zi+1 / m

= 81 /128

= **0.6328**

Bilangan acak 3

Zi+1 = (a . Zi + c) mod m

= (19 . 81 + 237) mod 128

= 1776 mod 128

= 112 (Random Number) R3 = Zi+1 / m

= 112 /128

= **0.875**

Yang terpenting adalah bahwa pembangkitan bilangan random menghasilkan R1, R2 … mendekati keseragaman dan independen. Pertimbangan harus pula diberikan kepada *maximum density* dan *maximum period*. *Maximum density* berarti bahwa nilai-nilai dihasilkan oleh Ri , i=1,2 … tidak memberikan gap yang cukup besar [0,1]. Ini dapat dicapai dengan pembuatan m yang besar. Untuk menghindari *cycling* (perulangan angka-angka yang dihasilkan) *generator* perlu mempunyai *possible periode* yang cukup besar. Ini biasa dicapai oleh pemilihan a,c,m, dan X0 yang sesuai.

# Pemograman Web

Tahap-tahap berikut ini merupakan pengertian teori dasar yang berhubungan dengan kasus yang akan dianalisis yaitu sebagai berikut :

# PHP

PHP *Hypertext Preprocessor* atau disingkat dengan PHP adalah suatu bahasa *scripting* khususnya digunakan untuk *web development.* Karena sifatnya yang *server side scripting*, maka untuk menjalankan PHP harus menggunakan web server.

PHP juga dapat diintegrasikan dengan HTML, JavaScript, JQuery, Ajax. Namun, pada umumnya PHP lebih banyak digunakan bersamaan dengan *file* bertipe HTML. Dengan menggunakan PHP Anda bisa membuat *website powerful* yang dinamis dengan disertai manajemen *database-*nya. Selain itu juga penggunaan PHP yang sebagian besar dapat jalan di banyak *platform,* menjadi salah satu alasan kenapa Anda harus menguasai PHP untuk menjadi *web*

*development* hebat.

Xampp merupakan paket server web PHP dan database MySQL, yang paling populer di kalangan pengembangan web menggunakan PHP dan MySQL sebagai Databasenya (Saiful & Ambarita, 2017).

# Sejarah PHP

Pada awalnya PHP meupakan kependekan dari *Personal Home Page (Situs personal*). PHP pertama kali dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted (FI),* yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari *web.*

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrograman yang tertarik untuk ikut mengembangkan PHP.

Pada November 1997, dirilis PHP/FI 2.0. Pada rilis 2.0 ini, *interpreter* PHP sudah diimplementasikan dalam program C. Di dalam rilis ini juga ikut disertakan modul-modul ekstensi yang meningkatkan kemampuan PHP/FI secara signifikan.

Kemudian pada tahun 1997, sebuah perusahaan bernama Zend menulis ulang *interpreter* PHP menjadi lebih bersih, lebih baik, dan lebih cepat. Kemudian pada Juni 1998, perusahaan tersebut merilis *interpreter* baru untuk PHP dan meresmikan rilis tersebut sebagai PHP 3.0 dan singkatan PHP diubah menjadi akronim berulang ***PHP: Hypertext Preprocessing****.*

Pada pertengahan tahun 1999, Zend merilis interpreter PHP baru dan rilis tersebut dikenal dengan PHP 4.0. PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya

untuk membangun aplikasi *web* kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi. Php adalah Salah satu aplikasi yang digunakan untuk memudahkan dalam melakukan pengelolaan database *Mysql* ( Buana, 2014).

# Kelebihan PHP

PHP berbasis *server side scripting.* PHP sendiri dapat melakukan tugas-tugas yang dilakukan dengan mekanisme CGI seperti mengambil, mengumpulkan data dari *database,* meng-*generate* halaman dinamis, atau bahkan menerima dan mengirim *cookie.* Dan yang menjadi keutamaan PHP itu sendiri adalah PHP bisa digunakan di berbagai *operating system,* diantaranya Linux, Unix, Windows, Mac OsX, RISC OS, dan *operating system* lainnya.

# Kekurangan PHP

Komunitas yang banyak tentu membawa kompetensi yang ketat. Para web *developer* yang menguasai PHP tiap hari semakin bertambah. Mudah dibajak karena sifat PHP yang merupakan *interpreter*, *source* code dari aplikasi PHP dapat dengan mudah dimodifikasi dan diubah fungsinya. Terkesan kurang *prestigious*, yakni mudah dipelajari oleh programmer pemula membuat bahasa pemograman PHP terkesan kurang *prestisius* jika dibandingkan dengan pemograman web lain yang terkesan lebih sulit untuk digunakan.

# Konsep Database

Adapun pengertian dan penjelasan tentang konsep database akan diuraikan dalam sub-sub berikut ini.

# Pengertian Database

Database adalah suatu aplikasi yang menyimpan sekumpulan data. Setiap database mempunyai API tertentu untuk membuat, mengakses, mengatur, mencari, dan menyalin data yang ada di dalamnya. Kumpulan data (elementer) yang secara logic berkaitan dalam mempresentasikan fenomena/fakta secara terstruktur dalam domain tertentu untuk mendukung aplikasi dalam system tertentu (N. P. Ade, 2016).

* + 1. ***MySQL***

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (Database management system) atau Relational Database Management System (RDBMS) DBMS yang Multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam database MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU General Public License (GPL), tetapi mereka

juga menjual di bawah lisensi komersial untuk kasus -kasus dimana penggunaannya tidak cocok dengan penggunaan GPL (Putra & Hadi, 2016).

Kelebihan dari MySQL adalah gratis, handal, selalu di-*update* dan banyak forum yang memfasilitasi para pengguna jika memiliki kendala. MySQL juga menjadi DBMS yang sering dibundling dengan *web server* sehingga proses instalasinya jadi lebih mudah.

4

# BAB III METODOLOGI PENELITIAN

* 1. **Kerangka Penelitian**

Berikut merupakan kerangka penelitian yang akan dilakukan:

Penelitian Pendahuluan

Pengumpulan Data

Analisa

Perancangan

Implementasi

Pengujian

# Gambar 3.1 Kerangka Penelitian

Gambar 3.1 merupakan kerangka penelitian yang dilakukan oleh peneliti, dimulai dari pendahuluan, pengumpulan data, analisa dan perancangan serta implementasi dan pengujian sistem secara bertahap dan berurutan. Adapun penjelasan lebih rinci dari tahapan-tahapan kerangka penelitian tersebut akan dibahas pada sub bab 3.2 di bawah ini.

# Tahapan Penelitian

Adapun tahapan - tahapan dalam penelitian ini adalah sebagai berikut:

# Penelitian Pendahuluan

Dalam aktifitas produksi pada *home-industry* keripik maco badarai Istiqomah ini, seringkali terjadi kesenjangan angka pendapatan setiap periodenya. Hal ini menyebabkan ketidakstabilan produksi seperti permintaan yang datang secara mendadak maupun stock barang yang terbatas yang menyebabkan aktifitas produksi tidak berjalan dengan baik.

Dengan menggunakan simulasi monte carlo mampu memberikan informasi untuk memprediksi pendapatan di *home-industry* keripik maco badarai Istiqomah, dimana produksi keripik maco pada *home-industry* keripik maco badarai Istiqomah menjadi lebih baik dengan adanya pemecahan masalah dengan *simulasi* metode monte carlo ini, sehingga dapat mengatasi masalah mengetahui keuntungan produk, membandingkan pendapatan disetiap periode, dan mengetahui tingkat minat Konsumen.

# Pengumpulan Data

Berikut tahapan untuk pengumpulan data adalah sebagai berikut.

# Waktu Penelitian

Adapun awal penelitian ini dilakukan pada Bulan Agustus 2018 dan berakhir pada bulan Februari 2019 yang dapat dilihat pada Tabel 3.1 Tabel berikut ini :

# Tabel 3.1 Tabel Penelitian

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Kegiatan | Agustus | Sep tember | Oktober | Novem Ber | Desem Ber | Januari | Februari |
| 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 |
| Perencanaan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengumpu lan Data |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Analisa |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
| Pembuatan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengujian |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Maintenan ce |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| laporan |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

# Tempat Penelitian

Penelitian ini dilakukan di *home-industry* Istiqomah yang beralamat di Jalan Sako ngalau no 13 Indarung, Kota Padang, Sumatera Barat.

# Metode Penelitian

* + - * 1. Penelitian Lapangan (*Field Research)*

Sebuah penelitian agar dapat menghasilkan data yang optimal, maka diperlukan penilitian lapangan, dimana penelitian lapangan ini melakukan pengambilan data secara langsung di *home-industry* Istiqomah indarung.

* + - * 1. Penelitian Perpustakaan ( *Library Research* )

Pengumpulan data-data melalui penelitian perpustakaan dengan membaca buku-buku dan referensi-referensi lainnya yaitu menyangkut masalah yang

berhubungan dengan pembahasan dalam skripsi ini, kemudian membandingkan dengan data yang telah diperoleh pada *field research.*

* + - * 1. Penelitian laboratorium (*Laboratory Research)*

Penelitian laboratorium berkaitan dengan *hardware* dan *software* yang digunakan dalam penelitian.

Adapun Perangkat keras (*hardware*) yang dipakai dalam penelitian ini, yaitu:

Merk : ASUS X450EA

Processor : AMD E1-2500 APU with Radeon(TM)

RAM : 2 Giga Byte

Hardisk : 512 Giga Byte

Flash disk : SanDisk 32 Gigabyte

Sedangkan Perangkat lunak (*software*) yang dipakai dalam penelitian ini, yaitu:

1. Sistem Operasi *Windows* 7.
2. Microsoft Office 2010
3. *XAMPP* v3.2.1
4. *MySQL* version 4.4.14
5. Rational Rose
6. Notepad++ v6.8.3
7. Mozila Firefox.

# Analisa

Untuk melakukan penelitian, penulis melakukan analisa terhadap data-data yang telah dikumpulkan serta menganalisa sistem yang akan dijalankan sebagai solusi dari perumusan masalah yang didapat. Setelah melakukan analisa, penulis juga melakukan perancangan terhadap sistem yang akan dibangun.

# Analisa Data

Analisa data dibutuhkan agar penulis dapat memahami secara keseluruhan mengenai objek penelitian serta pemahaman secara teoritis maupun praktis mengenai teknik dan metode yang akan digunakan dalam penyelesaian masalah-masalah yang telah dirumuskan pada *home-industry* Istiqomah.

# Analisa Proses

Setelah analisa data selanjutnya yaitu analisa proses, analisa proses dilakukan dengan simulasi menggunakan metode monte carlo. Analisa proses dibutuhkan agar penulis dapat menentukan tahapan demi tahapan pada perancangan sistem dan aplikasi yang akan digunakan.

# Analisa Sistem

Analisa sistem juga dibutuhkan agar penulis dapat mendeskripsikan alur kerja sistem berdasarkan kebutuhan penggunanya. Dengan adanya analisa sistem diharapkan dapat menciptakan suasana *user friendly* terhadap sistem yang akan dibangun.

# Perancangan

Setelah melakukan analisa terhadap data dan sistem, penulis juga melakukan perancangan terhadap sistem yang akan dibangun berdasarkan analisa sistem yang telah dilakukan.

# Perancangan Model

Adapun dalam melakukan perancangan sistem, penulis menggunakan *Unified Modeling Languange (UML)* sebagai *tools* dalam mendeskripsikan alur kerja sistem. Adapun penjelasan alur kerja sistem tersebut dideskripsikan dengan diagram- diagram *UML* diantaranya sebagai berikut :

1. *Usecase Diagram*

Diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor, dan digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi serta siapa saja yang berhak menggunakan fungsi-fungsi itu. Pada perancangan simulasi metode monte carlo di *home-industry* Istiqomah ini menggunakan satu aktor yang bisa mengoperasikan aplikasi yaitu manager. Adapun fasilitas yang diberikan untuk manager yaitu dapat mengoperasikan keseluruhan dari aplikasi tersebut.

1. *Class Diagram*

*Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel yang dimiliki oleh suatu kelas, dan operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas. Pada perancangan aplikasi simulasi penjualan ini

memiliki beberapa tabel sebagai penyimpanan data-data yang dibutuhkan dalam pengoperasiannya. Diagram ini akan memperlihatkan dengan jelas struktur dari setiap tabel yang ada serta menunjukkan hubungan antar tabel.

1. *Statechart Diagram*

*State Chart Diagram* akan menggambarkan cara memodelkan berbagai *state* (keadaan) keberadaan objek, digunakan untuk memodelkan lebih dinamis perilaku dari sistem.

1. *Activity Diagram*

*Activity Diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir.

* 1. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.
	2. *Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*).
	3. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.
1. *Sequence Diagram*

Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambar diagram sekuen maka 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.*

1. *Colaboration Diagram*

*Collaboration diagram* adalah perluasan dari objek diagram. Objek diagram menunjukkan objek-objek dan hubungannya satu dengan yang lain. *Collaboration diagram* menunjukkan *message-message* objek yang dikirimkan satu sama lain. Antara *collaboration diagram* dan *sequence diagram* bisa saling mengisi. Dengan demikian pada *collaboration diagram* kita bisa tambahkan nomor urut pada label sebuah *message* untuk menunjukkan urutan informasi. Titik dua (:) perlu digunakan untuk memisahkan nomor dengan *message.*

1. *Deployment Diagram*

Diagram *deployment* atau *deployment diagram* menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi. *Deployment diagram* menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, *server* atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lain yang bersifat fisikal.

# Implementasi

Tahap perancangan aplikasi setelah tahap analisis selesai. Implementasi ke dalam aplikasi merupakan tahap memasukkan data-data yang telah diolah dan menerapkan model yang terbaik sehingga data tersebut siap untuk dioperasikan. Dalam hal ini penulis merancang dan membangun sistem pemodelan dan simulasi berbasis web. Untuk mengimplementasikan suatu sistem yang telah dirancang, maka

diperlukan sebuah alat bantu komputer untuk mengoperasikan komputer itu sendiri yang memerlukan tiga buah komponen pendukung seperti *hardware, software, dan brainware*. Adapun fungsi dari masing-masing komponen dapat dijelaskan sebagai berikut :

1. Perangkat Keras (*Hardware*)

*Hardware* yang digunakan untuk merancang atau menjalankan program aplikasi yang telah dibuat dalam satu unit komputer yang lengkap dengan CPU, *harddisk* sebagai media penyimpanan data yang akan diinstalkan pada komputer.

1. Perangkat Lunak *(Software)*

Untuk menjalankan program aplikasi yang dirancang harus menggunakan beberapa *software* pendukung. *Software-software* pendukung yang harus dimiliki adalah mowes portable dan adobe dreamweaver yang berfungsi untuk merancang desain program aplikasi tersebut.

1. Manusia *(Brainware)*

*Brainware* merupakan operator yang berfungsi untuk mengoperasikan atau menjalankan sebuah sistem.

# Pengujian Sistem

Setelah tahap implementasi selesai, penulis juga melakukan pengujian yang bertujuan untuk melakukan koreksi terhadap sistem yang telah dibangun berdasarkan tingkat efisiensi sistem sebagai solusi dalam pemecahan masalah-masalah yang telah dirumuskan. Pengujian aplikasi merupakan tahap akhir dalam melakukan *testing,* guna untuk mengetahui kesalahan dalam sistem. Pengujian dilakukan dengan melihat apakah aplikasi tersebut sudah berjalan dengan benar dan sesuai dengan perancangan yang dilakukan.

**BAB IV**

**BIAYA DAN JADWAL KEGIATAN**

Penelitian dilakukan dalam jangka waktu 1 (satu) tahun dengan rincian kegiatan sebagaimana dijelaskan melalui matriks tabel berikut ini

**Tabel 4.1. Rencana Anggaran Biaya Penelitian**

|  |  |  |
| --- | --- | --- |
| **No** | **Jenis Pengeluaran** | **Biaya yang diusulkan (Rp)** |
| 1 | Gaji dan upah | Rp. 4.500.000,- |
| 2 | Peralatan Penunjang | Rp. 5.000.000,- |
| 2 | Bahan habis pakai | Rp. 5.500.000,- |
| 3 | Perjalanan | Rp. 3.000.000,- |
| 4 | Lain – lain (publikasi, seminar) | Rp. 12.000.000,- |

**4.2 Jadwal Penelitian**

Jadwal penelitian dapat dilihat pada *bar chart* dibawah ini :

**Tabel 4.2. Jadwal Penelitian**

|  |  |  |
| --- | --- | --- |
| **No** | **Jenis Kegiatan** | **Bulan** |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** |
| **I. Persiapan**  |  |
| 1  | Studi Pendahuluan  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2  | Mempelajari Literatur  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3  | Pengumpulan Data  |  |  |  |  |  |  |  |  |  |  |  |  |
| **II. Analisa dan Perancangan** |  |
| 4.  | Analisa Data  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5.  | Analisa Sistem  |  |  |  |  |  |  |  |  |  |  |  |  |