BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Penelitian Hondro (2018) mengenai aplikasi enkripsi dan dekripsi sms dengan algoritma *zig zag cipher* pada *mobile phone* berbasis *android*, disimpulkan bahwa algoritma *zig zag cipher* dapat digunakan untuk enkripsi dan dekripsi sms pada perangkat *android*. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* untuk kerahasiaan isi teks basis data.

Penelitian Hariati, dkk (2018) mengenai kombinasi algoritma *playfair cipher* dengan metode *zig-zag* dalam penyandian teks, disimpulkan bahwa algoritma *zig zag cipher* dapat dikombinasikan dengan algoritma *playfair cipher* sehingga mendapatkan kekuatan kerahasiaan data. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* dengan ROT13 untuk kekuatan kerahasiaan isi teks basis data.

Penelitian Rachmawati, dkk (2018) mengenai kombinasi algoritma *zig-zag cipher* dengan algoritma RSA dalam penyandian teks, disimpulkan bahwa algoritma *zig zag cipher* dapat dikombinasikan dengan algoritma RSA sehingga mendapatkan kekuatan penyandian teks. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* dengan ROT13 untuk kekuatan penyandian isi teks basis data.

Penelitian Rachmawati, dkk (2017) mengenai keamanan *file* dengan teknik *zig-zag* dan *huffman*, disimpulkan bahwa algoritma *zig zag cipher* mengamankan *file*

dengan merubah teks di dalamnya menjadi tidak dimengerti. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* demerubah isi teks basis data menjadi teks yang tidak dimengerti selain pihak yang berwenang.

Penelitian Firdaus dan Waluyo (2018) mengenai sistem keamanan *chat* berbasis *web* menggunakan algoritma *zig zag* studi kasus : noktournal motoshop disimpulkan bahwa algoritma *zig zag cipher* dapat mengamankan pesan teks *chatting* dengan merubah isi pesan teks *chatting* menjadi tidak dimengerti. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* demerubah isi teks basis data menjadi teks yang tidak dimengerti selain pihak yang berwenang.

Penelitian Hendrik (2020) mengenai kombinasi algoritma huffman dan algoritma ROT 13 dalam pengamanan file docx, disimpulkan bahwa ROT 13 mengamankan isi *file* Docx menjadi tidak dimengerti. Sedangkan penelitian penulis menggunakan metode *zig-zag cipher* demerubah isi teks basis data menjadi teks yang tidak dimengerti.

Penelitian Pramudya, dkk (2020) mengenai kombinasi algoritma ROT13 dan *vigenere cipher* pada alamat *directory file* untuk keamanan dokumen, disimpulkan bahwa kombinasi ROT 13 dan *vigenere cipher* dapat memperkuat kerahasiaan data. Sedangkan penelitian penulis menggunakan kombinasi metode *zig-zag cipher* dengan ROT13 sehingga dapat memperkuat kerahasiaan isi teks basis data.

Penelitian Yazid dan Haryanto (2020) mengenai perancangan aplikasi penyandian *file* teks dengan menggunakan algoritma ROT13 dan *rail fence cipher* disimpulkan bahwa kombinasi ROT 13 dan *rail fence cipher* dapat memperkuat

kerahasiaan data. Sedangkan penelitian penulis menggunakan kombinasi metode *zig-zag cipher* dengan ROT13 sehingga dapat memperkuat kerahasiaan isi teks basis data.

II.2. Landasan Teori

Landasan teori peneliti kutip dari beberapa teori yang berkaitan dengan penelitian ini yaitu :

II.2.1. Implementasi

Implementasi merupakan proses penterjemahan dari pemodelan menjadi bentuk aplikasi sesuai dengan kebutuhan informasi pengguna. Proses implementasi ada proses penterjemahan dari pemodelan ke dalam pengkodean atau pembentukan antarmuka. (Oktaviani dan Sauda, 2019 : 182).

II.2.2. Kriptografi

Kriptografi ialah ilmu yang mempelajari bagaimana cara menjaga supaya pesan atau data tetap aman pada saat dikirimkan dari pengirim ke penerima tanpa mengalami gangguan dari pihak ketiga. Kriptografi adalah ilmu dan seni untuk menjaga keamanan ketika pesan dikirim dari suatu tempat ke tempat lain. (Firdaus dan Waluyo, 2018 : 167).

II.2.2.1. Tujuan Kriptografi

Tujuan dari kriptografi adalah untuk tidak menyembunyikan keberadaan pesan , melainkan untuk menyembunyikan maknanya. Aspek keamanan yang diberikan kriptografi selain menyandikan pesan juga menyediakan beberapa aspek keamanan. Berikut aspek keamanan kriptografi :

1. Kerahasiaan (*confidentiality*), adalah layanan yang digunakan untuk menjaga isi pesan dari siapapun yang tidak berhak membacanya. Layanan ini direalisasikan dengan cara menyandikan pesan menjadi bentuk yang tidak dapat dimengerti. Misalnya pesan "Harap datang pukul 8" disandikan menjadi "TrxC#45motypetre!%".
2. Integritas data (*data integrity*), adalah layanan yang menjamin bahwa pesan masih asli / utuh atau belum pernah dimanipulasi selama pengiriman. Layanan ini direalisasikan dengan menggunakan tanda-tanda digital (*digital signature*). Pesan yang telah ditandatangani menyiratkan bahwa pesan yang dikirim adalah asli.
3. Otentifikasi (*authenification*), adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user authentification* atau *entity authentification*) maupun mengidentifikasi kebenaran sumber pesan (data origin authentification). Layanan ini direalisasikan dengan menggunakan *digital signature*.
4. Nirpenyangkalan (*non-repudiation*), adalah layanan untuk mencegah entitas yang berkomunikasi melakukan penyangkalan, yaitu pengirim pesan menyangkal

melakukan pengiriman atau penerima pesan menyangkal telah menerima pesan. (Hondro, 2019 : 122).

II.2.2.2. Komponen Kriptografi

Dalam melakukan pengamanan dengan ilmu kriptografi adapun komponen pendukung system kriptografi :

1. Pesan (*message*) adalah data atau informasi yang dapat dibaca atau dimengerti maknanya. Nama lainnya untuk pesan adalah plainteks (*plaintext*) atau teks jelas (*clear text*).
2. Pengirim (*sender*) adalah entitas yang melakukan pengiriman pesan kepada entitas lainnya.
3. Kunci (*cipher*)/*Secret Key* adalah aturan atau fungsi matematika yang digunakan untuk melakukan proses enkripsi dan dekripsi pada plaintext dan ciphertext.
4. *Ciphertext* adalah keluaran algoritma enkripsi. *Cipehertext* dapat dianggap sebagai pesan dalam bentuk tersembunyi. Algoritma enkripsi yang baik akan menghasilkan *ciphertext* yang terlihat teracak. Untuk selanjutnya digunakan istilah teks sandi sebagai padana kata *ciphertext*.
5. Enkripsi adalah mekanisme yang dilakukan untuk merubah *plaintext* menjadi *ciphertext*.
6. Dekripsi adalah mekanisme yang dilakukan untuk merubah *ciphertext* menjadi *plaintext*.
7. Penerima (*receiver*) adalah entitas yang menerima pesan dari pengirim/entitas yang berhak atas pesan yang dikirim. (Hondro, 2019 : 122).

II.2.2.3. Algoritma Kriptografi

Algoritma dalam kriptografi merupakan langkah-langkah logis bagaimana menyembunyikan pesan dari orang lain yang tidak berhak atas pesan tersebut.

1. Algoritma Enkripsi: Algoritma enkripsi memiliki 2 masukan teks asli dan kunci rahasia. Algoritma enkripsi melakukan transformasi terhadap teks asli sehingga menghasilkan teks sandi.
2. Algoritma Dekripsi: Algoritma dekripsi memiliki 2 masukan yaitu teks sandi dan kunci rahasia. Algoritma dekripsi memulihkan kembali teks sandi menjadi teks asli bila kunci rahasia yang dipakai algoritma dekripsi sama dengan kunci rahasia yang dipakai algoritma enkripsi.
3. Algoritma Kunci (*Key*): Didalam Kutipan Bahan Perkuliahan Sistem Kriptografi Ir. Rinaldi Munir, M.T., Kunci adalah yang dipakai untuk melakukan enkripsi dan dekripsi. Kunci terbagi dua bagian yaitu kunci rahasia (*private key*) dan kunci umum (*public key*). (Hondro, 2019 : 122).

II.2.2.4. Jenis Kriptografi

Berdasarkan perkembangan dari tahun ke tahun sejak pertama kali kriptografi ditemukan, ada dua jenis algoritma kriptografi, yaitu:

1. Kriptografi Klasik

   Algoritma kriptografi yang termasuk kedalam kriptografi klasik ini digunakan sejak sebelum era komputerisasi dan kebanyakan menggunakan teknik kunci simetris. Metode menyembunyikan pesannya adalah dengan teknik substitusi atau

transposisi atau keduanya. Teknik substitusi bekrja dengan cara menggantikan karakter dalam plaintext menjadi karakter lain yang hasil akhirnya adalah ciphertext. Sedangkan transposisi merupakan teknik untuk mengubah plaintext menjadi ciphertext dengan cara permutasi karakter. Kombinasi dari keduanya sebenarnya yang secara kompleks yang melatar belakangi terbentuknya berbagai macam algoritma kriptografi modern. Salah satu kriptografi klasik adalah zig-zag cipher dan ROT13.

a. Metode Zig-Zag Cipher

Metode *zig-zag cipher* merupakan salah satu algoritma kriptografi klasik dengan teknik transposisi. Teknik transposisi menggunakan permutasi karakter, yang mana dengan menggunakan teknik ini pesan yang asli tidak dapat dibaca kecuali orang yang memiliki kunci untuk menggembalikan pesan tersebut ke bentuk semula. Algoritma *zig-zag* ini merupakan pembentukan dari algoritma transposisi kolom (*Columnar Transposition Cipher*) dan Transposisi *Rail Fance Cipher*. Teknik transposisi menggunakan permutasi karakter, sehingga pesan yang asli tidak dapat dibaca kecuali orang yang memiliki kunci untuk menggembalikan pesan tersebut ke bentuk semula. Metode *zig-zag* memiliki kelebihan dibandingkan dengan algoritma lainnya (dalam *cipher* transposisi), dalam proses penulisan plainteks menjadi chiperteks karena penulisan dapat dilakukan dari baris mana saja. Hal ini akan menambah kerumitan dalam proses enkripsi maupun dekripsi. (Hariati, dkk, 2018 : 14).

Berikut ini akan diuraikan contoh enkripsi berdasarkan algoritma *zig-zag cipher*.

1. Plaintext = DENI ANGGARA

   Kunci Zig-zag = 3 dan offset = 0

D								**G**			
	E		**I**		**A**		**G**		**A**		**A**
		N				**N**				**R**	

Proses penulisan cipher dimulai dari baris pertama, kemudian disusul oleh baris berikutnya. Karakter-karakter yang ada pada setiap baris dijadikan sebagai cipher. Berasarkan tabel transposisi di atas, maka ditemukan ciphertext D GEIAGAANNR.

Berikut ini akan diuraikan contoh dekripsi berdasarkan algoritma *zig-zag cipher*.

Ciphertext = D GEIAGAANNR

Kunci Zig-zag = 3 dan offset = 0 dan jumlah cipher = 12

x				**x**				**x**			
	x		**x**		**x**		**x**		**x**		**x**
		x				**x**				**x**	

Susun berdasarkan kotak ketentuan zig-zag cipher sehingga menjadi seperti berikut :

D								**G**			
	E		**I**		**A**		**G**		**A**		**A**
		N				**N**				**R**	

Susun menurun kebawah sehingga menjadi *plaintext* : DENI ANGGARA.

2. Plaintext = SELAMAT PAGI

   Kunci Zig-zag = 3 dan offset = 0

S				**M**				**P**			
	E		**A**		**A**				**A**		**I**
		L				**T**				**G**	

Proses penulisan cipher dimulai dari baris pertama, kemudian disusul oleh baris berikutnya. Karakter-karakter yang ada pada setiap baris dijadikan sebagai cipher. Berasarkan tabel transposisi di atas, maka ditemukan ciphertext SMPEAA AILTG. Berikut ini akan diuraikan contoh dekripsi berdasarkan algoritma *zig-zag cipher*.

Ciphertext = SMPEAA AILTG

Kunci Zig-zag = 3 dan offset = 0 dan jumlah cipher = 12

x				**x**				**x**			
	x		**x**		**x**		**x**		**x**		**x**
		x				**x**				**x**	

Susun berdasarkan kotak ketentuan zig-zag cipher sehingga menjadi seperti berikut :

S				**M**				**P**			
	E		**A**		**A**				**A**		**I**
		L				**T**				**G**	

Susun menurun kebawah sehingga menjadi *plaintext* : SELAMAT PAGI.

3. Plaintext = SELAMAT SORE

Kunci Zig-zag = 3 dan offset = 0

S				**M**				**S**			
	E		**A**		**A**				**O**		**E**
		L				**T**				**R**	

Proses penulisan cipher dimulai dari baris pertama, kemudian disusul oleh baris berikutnya. Karakter-karakter yang ada pada setiap baris dijadikan sebagai cipher. Berasarkan tabel transposisi di atas, maka ditemukan ciphertext SMSEAA OELTR.

Berikut ini akan diuraikan contoh dekripsi berdasarkan algoritma *zig-zag cipher*.

Ciphertext = SMSEAA OELTR

Kunci Zig-zag = 3 dan offset = 0 dan jumlah cipher = 12

x				**x**				**x**			
	x		**x**		**x**		**x**		**x**		**x**
		x				**x**				**x**	

Susun berdasarkan kotak ketentuan zig-zag cipher sehingga menjadi seperti berikut :

S				**M**				**S**			
	E		**A**		**A**				**O**		**E**
		L				**T**				**R**	

Susun menurun kebawah sehingga menjadi *plaintext* : SELAMAT SORE.

4. Plaintext = SANGAT MUDAH

Kunci Zig-zag = 3 dan offset = 0

S				**A**				**U**			
	A		**G**		**T**		**M**		**D**		**H**
		N								**A**	

Proses penulisan cipher dimulai dari baris pertama, kemudian disusul oleh baris berikutnya. Karakter-karakter yang ada pada setiap baris dijadikan sebagai cipher. Berasarkan tabel transposisi di atas, maka ditemukan ciphertext SAUAGTMDHN A.

Berikut ini akan diuraikan contoh dekripsi berdasarkan algoritma *zig-zag cipher*.

Ciphertext = SAUAGTMDHN A

Kunci Zig-zag = 3 dan offset = 0 dan jumlah cipher = 12

x				**x**				**x**			
	x		**x**		**x**		**x**		**x**		**x**
		x				**x**				**x**	

Susun berdasarkan kotak ketentuan zig-zag cipher sehingga menjadi seperti berikut :

S				**A**				**U**			
	A		**G**		**T**		**M**		**D**		**H**
		N								**A**	

Susun menurun kebawah sehingga menjadi *plaintext* : SANGAT MUDAH.

5. Plaintext = ITU ITU SAJA

Kunci Zig-zag = 3 dan offset = 0

I				**I**				**S**			
	T				**T**				**A**		**A**
		U				**U**				**J**	

Proses penulisan cipher dimulai dari baris pertama, kemudian disusul oleh baris berikutnya. Karakter-karakter yang ada pada setiap baris dijadikan sebagai cipher. Berasarkan tabel transposisi di atas, maka ditemukan ciphertext IIST T AAUUJ.

Berikut ini akan diuraikan contoh dekripsi berdasarkan algoritma *zig-zag cipher*.

Ciphertext = IIST T AAUUJ

Kunci Zig-zag = 3 dan offset = 0 dan jumlah cipher = 12

x				**x**				**x**			
	x		**x**		**x**		**x**		**x**		**x**
		x				**x**				**x**	

Susun berdasarkan kotak ketentuan zig-zag cipher sehingga menjadi seperti berikut :

I				**I**				**S**			
	T				**T**				**A**		**A**
		U				**U**				**J**	

Susun menurun kebawah sehingga menjadi *plaintext* : ITU ITU SAJA.

b. Metode ROT13

ROT13 (Rotate 13) adalah enkripsi substitution cipher yang umum digunakan di sistem operasi UNIX. Pada sistem enkripsi ROT13 sebuah huruf digantikan dengan huruf yang letaknya di atas 13 posisi darinya. Enkrip metode ROT13 menggunakan penjumlahan *ascii plaintext* dengan *ascii* kunci 13 sebagai berikut :

$C_i = P_i + 13$

Keterangan :

C_i = *Ciphertext* hasil enkrip

P_i = *Plaintext* yang akan dienkrip

Dekrip metode ROT13 menggunakan pengurangan *ascii ciphertext* dengan *ascii* kunci 13 sebagai berikut :

$P_i = C_i - 13$

Keterangan :

C_i = *Ciphertext* yang akan didekrip

P_i = *Plaintext* hasil dekrip. (Yazid dan Haryanto, 2021 : 68).

2. Kriptografi Modern

Algoritma kriptografi yang termasuk ke dalam jenis kriptografi modern dengan memiliki tingkat kesulitan yang kompleks, dan kekuatan kriptografinya ada pada key atau kuncinya. Kriptografi modern menggunakan pengolahan simbol biner karena berjalan mengikuti operasi komputer digital, sehingga membutuhkan dasar berupa pengetahuan terhadap matematika untuk bias mempelajari dan menguasainya. (Ziaurrahman, dkk, 2019:64).

II.2.5. Kerahasiaan

Rahasia adalah suatu yang disembunyikan dan hanya diketahui oleh satu orang, oleh beberapa orang saja, atau kalangan tertentu. Kerahasiaan merupakan pembatasan pengungkapan informasi pribadi tertentu. Dalam hal ini mencangkup

tanggung jawab untuk menggunakan, mengungkapan, atau mengeluarkan informasi hanya dengan sepengetahuan dan izin individu. (Siswati, 2019 : 93).

II.2.6. Teks

Teks merupakan satuan bahasa yang mengandung pikiran dengan struktur yang lengkap. Teks diartikan sebagai sebuah satuan bahasa. Hanya saja satuan bahasa yang dimaksud bukan satuan bahasa gramatikal seperti klausa atau kalimat dan tidak ditentukan oleh ukurannya. Satuan bahasa yang digunakan adalah yang lengkap secara tertulis seperti buku, surat, dokumen tertulis dan lain sebagainya. (Yani, dkk, 2020 : 2).

II.2.7. Basis Data

Basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basis data tersebut. Basis data adalah representasi kumpulan fakta yang saling berhubungan disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan. Basis data merupakan sekumpulan informasi yang saling berkaitan pada suatu subjek tertentu pada tujuan tertentu pula. Basis data adalah susunan record data operasional lengkap dari suatu organisasi atau perusahaan, yang diorganisir dan disimpan secara terintegrasi dengan menggunakan metode tertentu dalam komputer

sehingga mampu memenuhi informasi yang optimal yang dibutuhkan oleh para pengguna. (Helmund, 2021 : 81).

Basis data bisa diartikan juga sebagai sekumpulan data yang disusun dalam bentuk beberapa tabel yang saling memiliki relasi maupun berdiri sendiri". Berdasarkan beberapa definisi diatas, menurut penulis database merupakan kumpulan data yang berbentuk tabel dan memiliki hubungan antar tabel. Pada perancangan database yang digunakan dalam penelitian ini adalah metode data base life cycle yang artinya database tersebut akan terus berputar sesuai dengan situasi yang terus menerus berkembang. (Shabrina, dkk, 2021 : 100).

II.2.8. *Visual Basic* 2010

Visual basic 2010 adalah inkarnasi dari bahasa *Visual Basic* yang sangat populer dan telah dilengkapi dengan fitur serta fungsi yang setara dengan bahasa tingkat tinggi lainnya seperti C++. *Visual Basic* dapat digunakan untuk membuat aplikasi *Windows*, *mobile*, *web*, dan *office* yang kompleks dengan menggunakan kode yang anda tulis, atau kode yang telah ditulis oleh orang lain dan kemudian dimasukkan ke dalam program. (Putri dan Putra, 2018 : 52).

II.2.11. *Unified Modeling Language* (UML)

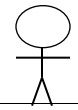
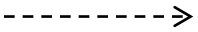
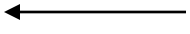
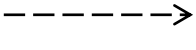
Unified Modeling Language (UML) adalah sebuah bahasa untuk menentukan, visualisasi, kontruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat

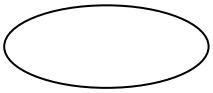
lunak. *Artifact* dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya.

1. *Use Case Diagram*

Use Case Diargam merupakan pemodelan untuk melakukan (*behavior*) sistem informai yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Berikut adalah simbol-simbol yang ada pada diagram *use case*. *Use case diagram* dapat digambarkan dengan sumber-sumber pada Tabel II.1.

Tabel II.1. Simbol *Use Case*

Notasi	Keterangan	Simbol
Actor	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.	
Depedency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).	
Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).	
Include	Menspesifikasikan bahwa *use case* sumber secara eksplisit.	
Extend	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.	
Association	Apa yang mnghubungkan antara objek satu dengan objek lainnya.	

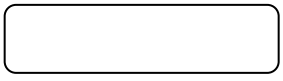
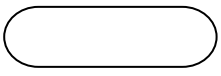
System	Menspesifikan paket yang menampilkan sistem secara terbatas.	
Use Case	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu *actor*.	
Collaboration	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).	
Note	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi	

(Sumber : Pitrawati dan Sanjaya, 2021 : 157)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Perlu diperhatikan bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Berikut adalah simbolsimbol yang ada pada diagram aktivitas. *Activity diagram* dapat digambarkan dengan simbol-simbol seperti pada tabel II.2.

Tabel II.2. Simbol *Activity Diagram*

Notasi	Keterangan	Simbol
Activity	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.	
Action	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.	
Initial Node	Bagaimana objek dibentuk atau diawali.	

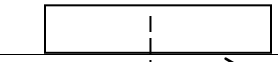
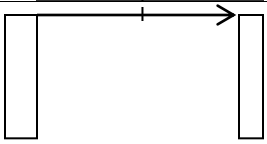
Activity Final	Bagaimana objek dibentuk dan dihancurkan	
Fork Node	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran	

(Sumber : Pitrawati dan Sanjaya, 2021 : 157)

3. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence Diagram* dapat digambarkan dengan simbol-simbol seperti pada Tabel II.3.

Tabel II.3. Simbol *Sequence Diagram*

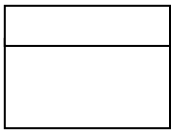
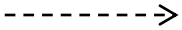
Nama	Keterangan	Gambar
Lifeline	Objek *entity*, antarmuka yang saling berinteraksi.	
Message	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.	
Message	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi.	

(Sumber : Pitrawati dan Sanjaya, 2021 : 157)

4. *Class Diagram* (Diagram Kelas)

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. Berikut adalah simbol-simbol yang ada pada diagram Kelas. *Class diagram* dapat digambarkan dengan simbol-simbol seperti pada Tabel II.4.

Tabel II.4. *Class Diagram*

Notasi	Keterangan	Gambar
Generalization	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).	
Nary Association	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.	
Class	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.	
Collaboration	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.	
Realization	Operasi yang benar-benar dilakukan oleh suatu objek.	
Depedency	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (*independent*) akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri	
Assocation	Apa yang menghubungkan antara objek satu dengan objek lainnya	

(Sumber : Pitrawati dan Sanjaya, 2021 : 157)