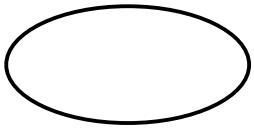
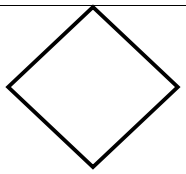
Sedangkan menurut Mata-Toledo dan Cushman dalam (Rusmawan, 2019) mendefinisikan bahwa "Entity Relationship Diagram (ERD) merupakan representasi grafis dari logika database dengan menyertakan deskripsi detail mengenai seluruh entitas (*entity*), hubungan (*relationship*), dan batasan (*constraint*)".

Bisa disimpulkan bahwa ERD sangat di butuhkan untuk menganalisis dalam penyelesaian pengembangan sistem untuk mendesain suatu model data secara detail dari seluruh entitas (*entity*), hubungan (*relationship*), dan batasan (*constraint*). Berikut symbol-simbol yang sering di gunakan antaralain sebagai berikut:

Tabel II.1.

Simbol-simbol *Entity Relationship Diagram*

Simbol	Keterangan
Entitas 	Entitas merupakan data inti yang akan disimpan, bakal tabel pada basis data, benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer. Penamaan entitas biasanya lebih ke kata benda dan belum merupakan nama tabel.
Atribut 	Field atau kolom data yang butuh disimpan dalam suatu entitas.
Relasi 	Relasi yang menghubungkan antar entitas, biasanya diawali dengan kata kerja.
Assosiasi	Penghubung antara relasi dan entitas dimana di kedua ujungnya memiliki multiplicity kemungkinan jumlah pemakaian. Kemungkinan jumlah maksimum keterhubungan antara entitas satu dengan

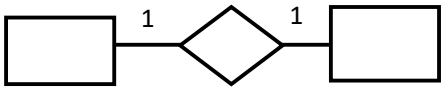
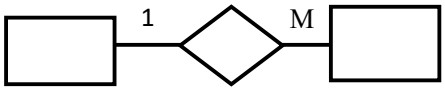
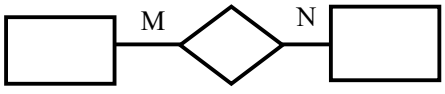
	entitas yang lain disebut dengan kardinalitas. Misalkan ada kardinalitas 1 ke N atau sering disebut dengan one to many menghubungkan entitas A dan entitas B.

Sumber: (Rusmawan, 2019)

Menurut (Andriani & Purnama, 2019), "*Entity Relationship Diagram* (ERD) memiliki hubungan antara dua atau lebih entitas yang saling berkaitan yang terbentuk dari jumlah maksimal entitas yang disebut sebagai kardinalitas". Jenis-jenis kardinalitas sebagai berikut:

Table II.2

Simbol-simbol kardinalitas

Simbol	Keterangan
	**Satu ke Satu atau One-To-One (1 - 1)** Entitas pada himpunan entitas A paling banyak hanya membentuk satu hubungan dengan entitas pada himpunan entitas B yang karidnalitasnya disimbolkan dengan 1 – 1.
	**Satu ke Banyak atau One-To- Many (1 – M)** Entitas pada himpunan entitas A dapat membentuk hubungan dengan banyak entitas pada himpunan entitas B yang kardinalitasnya disimbolkan dengan 1 – M.
	**Banyak ke Banyak atau Many- To-Many (M – N)** Entitas pada himpunan entitas A dapat membentuk hubungan dengan banyak entitas pada himpunan entitas B demikian pula

	sebaliknya yang kardinalitasnya disimbolkan dengan M – N.

Sumber: (Andriani & Purnama, 2019)

4.1.2 *Unified Modeling Language (UML)*

Menurut Gata & Windu dalam (Hendini, 2016) "*Unified Modeling Language (UML*) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak". untuk mengembangkan suatu sistem berorientasi objek perlu menggunakan metodologi UML untuk mendukung pengembangan sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML, adalah sebagai berikut:

1. *Use Case Diagram*

Menurut (Sukamto & Shalahuddin, 2018) "*Use case* diagram merupakan diagram atau pemodelan yang yang mendeskripsikan antar aktor dengan kelakuan (*behavior*) sistem informasi yang akan dibuat".

2. *Activity Diagram*

Menurut (Sukamto & Shalahuddin, 2018) “Diagram aktivitas atau *activity diagram* merupakan bentuk visual dari alur *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak yang berisi kegiatan dan tindakan, serta terdapat juga pilihan dan, pengulangan”. Diagram aktivitas menggambarkan kegitan yang dilakukan oleh sistem.

3. *Sequence Diagram*

Menurut (Sukamto & Shalahuddin, 2018) menjelaskan bahwa “Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek”.

4. *Class Diagram*

Menurut (Indrajani, 2015) “Diagram ini di gunakan untuk menggambarkan perbedaan yang mendasar antara class-class, hubungan antar-class, dan di mana sub-sistem class tersebut”. Class diagram mempunyai nama class, attributes, operations, serta association (hubungan antar-class).

Sedangakan menurut (Abdulghani et al., 2017) “Tidak hanya mendesain visualisasi dan mendokumentasikan yang ada dalam sistem, class diagram juga bisa mengesekusi kode dalam software yang akan di bangun”.

Setiap class memiliki atribut dan method yang menggambarkan atau mendifinisikan class tersebut (Abdulghani et al., 2017). Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis sebagai berikut:

4.1.3 *Logical Record Structure (LRS)*

Menurut Dhanta dalam (Junianto & Primaesha, 2015) "LRS (*Logical Record Structure*) adalah representasi dari struktur record-record pada tabel-tabel yang terbentuk dari hasil antar himpunan entitas, menentukan kardinalitas, jumlah table dan *Foreign Key* (FK)"

Sedangkan menurut ASFA dalam (Imaniawan & Elsa, 2017) "LRS merupakan transformasi dari penggambaran ERD dalam bentuk yang lebih jelas dan mudah untuk dipahami". *Logical Record Structure* seperti penggambaran pada normalisasi file, tetapi tidak digambarkan simbol asterisk (*) sebagai *simbol primary key* dan *foreign key.*

4.1.4 *Adobe XD*

Adobe XD atau bisa disebut dengan Adobe Experience Design CC adalah "Aplikasi untuk membuat suatu desain berfokus pada pengalaman pengguna yang dikembangkan dan diterbitkan oleh Adobe Systems Adobe XD mendukung desain vektor dan wireframing, dan menciptakan prototipe interaktif sederhana" (Rahman et al., 2020). Software ini lebih fokus kepada desain dan pengembangan produk digital agar lebih mudah dan ringkas.