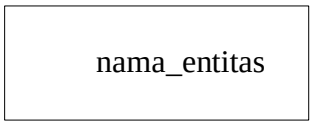
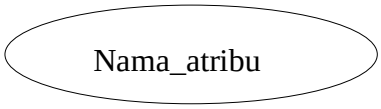
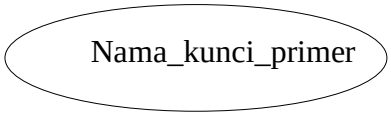
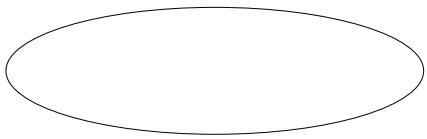
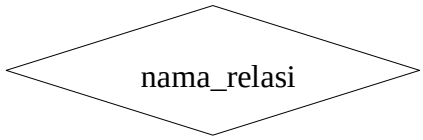
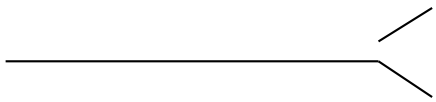
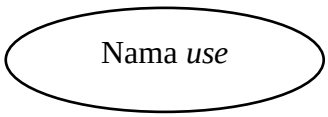
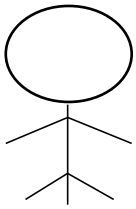
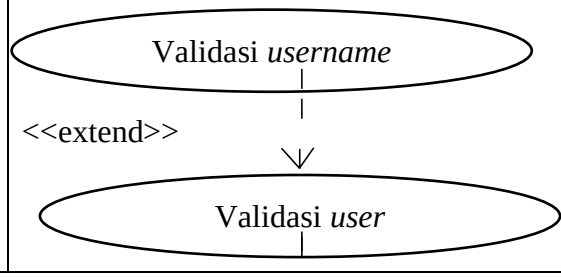
DAFTAR SIMBOL

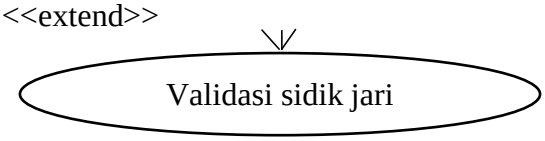
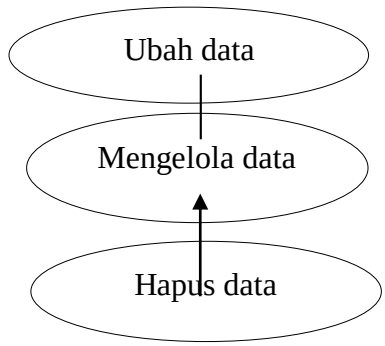
A. Simbol *Entity Relationship Diagram* (ERD)

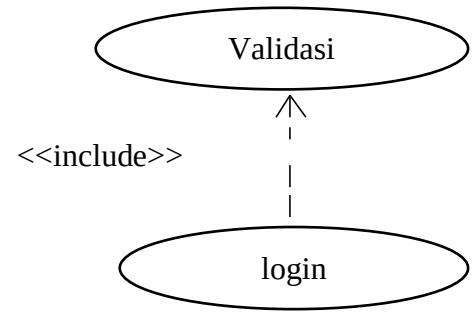
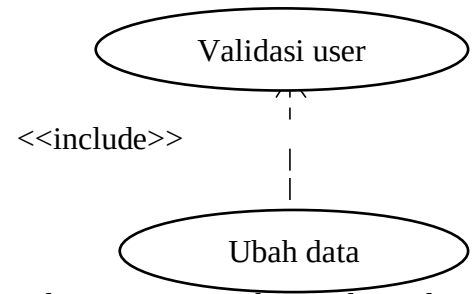
Simbol	Deskripsi
Entitas / *entity* 	Entitas merupakan data inti yang akan disimpan; bakal tabel pada basis data; benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer;penamaan entitas biasannya lebih ke kata benda dan belum merupakan nama tabel
Atribut 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas
Atribut Kunci Primer 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan; biasannya berupa id; kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat

	unik(berbeda tanpa ada yang sama)
Atribut multinilai/*multivalue* 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu
Relasi 	Relasi yang menghubungkan antar entitas; biasannya diawali dengan kata kerja
Asosiasi/*association* 	Penghubung antara relasi dan entitas di mana 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

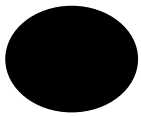
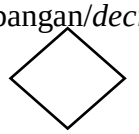
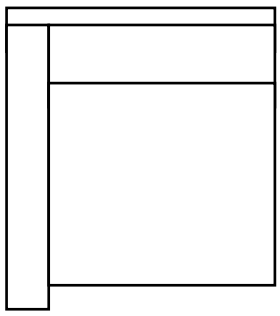
B. Simbol *Use Case Diagram*

No	Simbol	Deskripsi
1.	*Use Case* 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawala frase nama *use case*.
2.	Aktor/*actor* 	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3.	Assosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpatisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4.	Extensi/*extend* 	Relasi *use case* tambahan ke sebuah *use case* di mana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrogaman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misalnya 

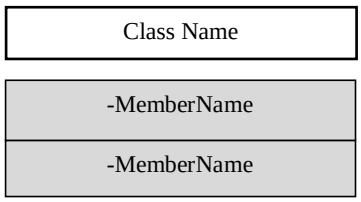
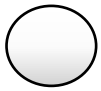
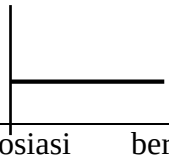
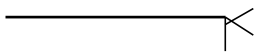
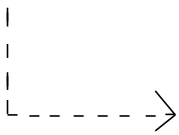
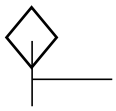
		<<extend>>  Arah panah mengarah pada *use case* yang ditambahkan, biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
5.	Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).
6.	Menggunakan/*include*/*uses* <<include>> 	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. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case*: - Include berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, missal

	<<uses>>	pada kasul berikut:  - *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan di jalankan, missal pada kasus berikut:  Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

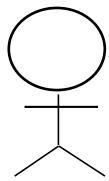
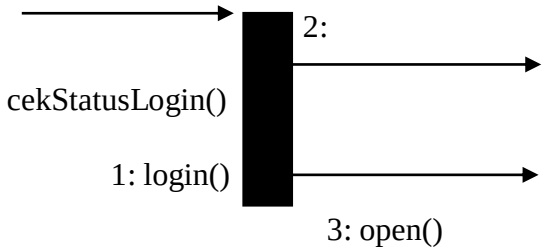
C. Simbol *Activity* Diagram

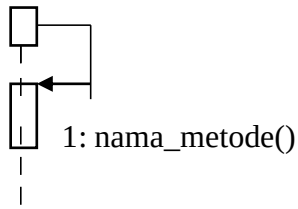
No.	Simbol	Deskripsi
1.	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.	Aktivitas	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.	Penggabungan/*join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.	Status akhir	Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

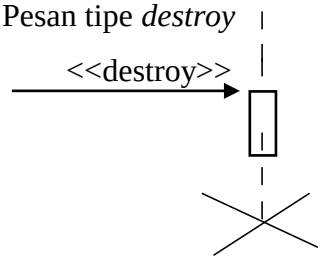
D.*Class* Diagram

No.	Simbol	Deskripsi
1.	Kelas 	Kelas pada struktur sistem.
2.	Antarmuka/*interface* 	Sama dengan konsep interface dalam pemrogaman berorintasi objek.
3.	Asosiasi/*association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
4.	Asosiasi berarah/*directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).
6.	Kebergantungan/*dependensi* 	Relasi antar kelas dengan makna kebergantungan antar kelas.
7.	Agrasi/*aggregation* 	Relasi antar kelas dengan makna semua bagian (*whole part*).

E. *Sequence* Diagram

No	Simbol	Deskripsi
1.	Aktor  Atau Nama Aktor Tanpa waktu aktif	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan dalam menggunakan kata benda diawal frase nama aktor.
2.	Garis hidup/*lifeline* 	Menyatakan kehidupan suatu objek.
3.	Objek Nama objek : Nama	Menyatakan objek yang berinteraksi pesan.
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya: 

		Maka cekStatusLogin() dan open() dilakukan didalam metode login(). Aktor tidak memiliki waktu aktif.
5.	Pesan tipe *create* <<create>>	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe *call* 1: nama_metode()	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
7.	Pesan tipe *send* 1: masukkan	Menyatakan bahwa suatu objek mengirimkan data/masukkan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
8.	Pesan tipe *return* 1: keluaran	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau

		metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9.	Pesan tipe *destroy*  	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang yang diakhiri, sebaliknya jika ada *create* maka ada *destroy*.