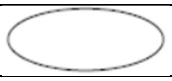
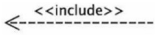
Interview adalah teknik yang tepat utuk menemukan *use case*. Setelah *use case* berhasil diidentifikasi, penting untuk mencatat kondisi sebelum dan sesudah *use case* dilakukan. Hasil dari wawancara akan dirangkum dalam daftar calon *class*. Hal ini akan menjadi dasar untuk berbicara dengan pengguna. Langkah yang bagus apabila *interview* dilakukan pada sekumpulan pengguna. Tujuan akhirnya adalah mengidentifikasi kandidat *use case* dan aktor [14].

Table 1 Simbol Use Case Diagram [15]

Gambar	Keterangan
	*Use case* merepresentasikan fungsi sistem sebagai unit yang saling berinteraksi, biasanya digambarkan dalam bentuk kata kerja.
	Aktor merupakan representasi dari individu atau sistem lain yang menjalankan fungsi dalam sistem target. Untuk mengidentifika sikan aktir, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa Aktor berperan dalam interaksi dengan *use case*, namun tidak mengatur jalannya *use case* tersebut.
	Hubungan antara aktor dan *use case* digambarkan sebagai garis lurus tanpa panah, menunjukkan siapa yang memulai interaksi, bukan jenis data yang terlibat.
	Hubungan antara aktor dan *use case* yang digambarkan dengan panah terbuka menunjukkan adanya interaksi pasif dari aktor terhadap sistem.
	*Include* menunjukkan bahwa suatu *use case* adalah bagian wajib dari *use case* lainnya, seperti pemanggilan fungsi dalam sebuah program.
	*Extend* digunakan untuk menggambarkan tambahan perilaku pada *use case* jika kondisi tertentu dipenuhi.

2. *Activity Diagram*

Activity Diagram merupakan elemen krusial dalam UML yang merepresentasikan sisi dinamis dari sebuah sistem. *Activity* diagram dapat secara mudah digunakan untuk menggambarkan logika prosedur, proses bisnis, serta alur kerja dalam sebuah sistem bisnis. *Activity* diagram mempunyai peran seperti halnya *flowchart*, akan tetapi Berbeda dengan *flowchart*, diagram aktivitas mampu merepresentasikan proses yang berjalan secara paralel, sedangkan *flowchart* tidak memiliki kemampuan tersebut. *Activity diagram* bertujuan merekam perilaku dinamis dari sistem dengan memperlihatkan alur komunikasi antar aktivitas. Secara umum, *activity diagram* digunakan untuk menggambarkan aliran aktivitas, urutan aktivitas, serta paralelisme, percabangan, dan aliran konkuren dalam sistem [14].

Table 2 Simbol-simbol Activity Diagram [14]

Gambar	Keterangan
	Titik awal
	Titik akhir
	*Activity*
	Pilihan untuk pengambilan keputusan
	Fork ; digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu
	*Rake* ; menunjukan adanya dekomposisi
	Tanda waktu
	Tanda pengiriman
	Tanda penerimaan
	Aliran akhir (*Flow Final*)

2.2. *Entity Relationship Diagram* (ERD)

Entity relationship adalah salah satu bagaimana tahapan perancang basis data untuk mengembangkan sebuah perangkat lunak pada sistem basis data yang didasarkan pada hasil perancang model konseptual dan relational. *Entity Relationship Diagram* (ERD) adalah diagram yang berbentuk notasigrafis yang berada dalam pembuatan *database* menghubungkan antara data satu dan data yang lain. Fungsi ERD adalah sebagai alat bantu dalam pembuatan *database* dan memberikan gambaran bagaimana kerja *database* yang akan dibuat [16].