BAB II
LANDASAN TEORI

II.1. Sistem Informasi

Sistem informasi adalah sekumpulan elemen yang saling bekerja sama baik secara manual atau berbasis komputer yang didalamnya ada pengumpulan, pengolahan, pemprosesan dan penyimpanan data untuk mengasilkan informasi yang berguna untuk proses pengambilan keputusan [8].

II.2. Komponen Sistem Informasi

Komponen sistem informasi terdiri dari beberapa hal sebagai berikut [9]:

1. Blok Masukan

   Masukan disini mewakili data yang masuk ke dalam sistem informasi. Masukan disini termasuk metode-metode dan media untuk menangkap data yang akan dimasukan, yang dapat berupa dokumen-dokumen dasar .

2. Blok Model

   Blok ini terdiri dari prosedure, logika dan model matematik yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cara yang sudah ditentukan untuk menghasilkan keluaran yang diinginkan.

3. Blok Keluaran

   Blok keluaran menghasilkan informasi yang berkualitas dan dokumentasi yang berguna untuk semua tingkatan manajemen serta semua pemakai sistem.

4. Blok Teknologi

   Teknologi merupakan “kotak alat” dalam sistem informasi. Teknologi digunakan untuk menerima masukan, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran dan membantu pengendalian dari sistem secara keseluruhan.

5. Blok Basis Data

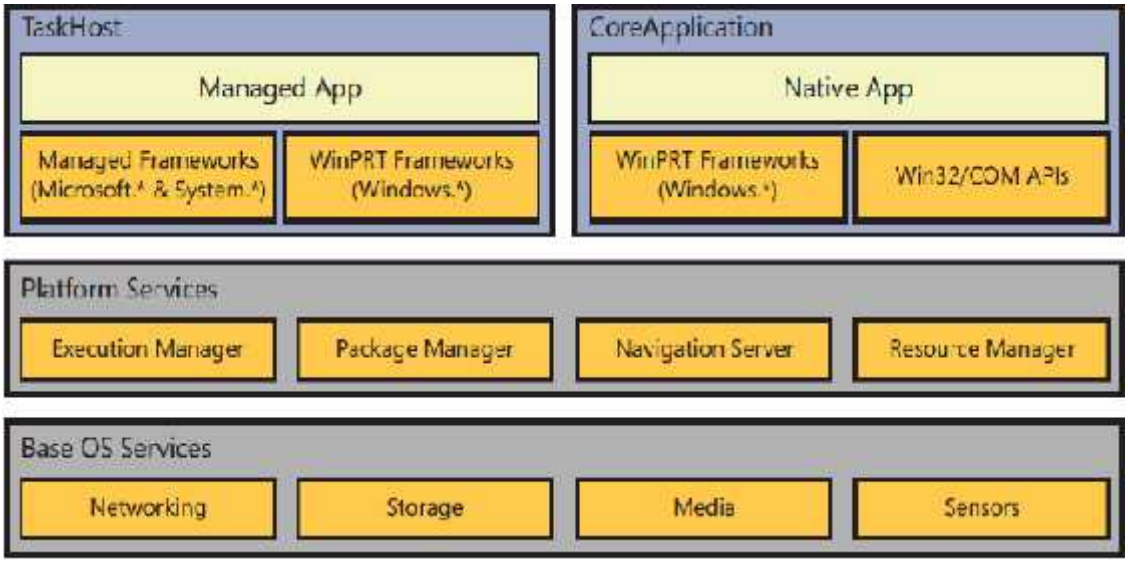
   Basis data merupakan kumpulan dari data yang saling berhubungan satu dengan lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.

6. Blok Kendali

   Banyak hal yang dapat merusak sistem informasi, seperti misalnya bencana alam, api, temperature, air, debu, kecurangan, kegagalan sistem, kesalahan-kesalahan, ketidak efisiensian, sabotase dan lain sebagainya.

II.3. Windows Phone

Windows Phone adalah serangkaian sistem operasi yang dikembangkan oleh Microsoft dan merupakan penerus dari *platform Windows Mobile*. Sistem operasi ini diluncurkan bulan Oktober 2010 dan diluncurkan di Asia pada awal tahun 2011. Versi terakhir dari sistem operasi ini adalah Windows Phone 8.1 yang diluncurkan bulan April 2014. Microsoft menciptakan *user interface* baru pada Windows Phone dengan bahasa desain yang disebut *modern design language*. Dalam sistem operasi Windows Phone terdapat arsitektur seperti pada Gambar II.1.



Gambar II.1 Arsitektur Sistem Operasi Windows Phone

Di bagian atas dari *stack* terdapat dua model aplikasi yang berbeda. Kotak yang berlabel "*TaskHost*" mewakili model aplikasi XAML, yang telah menjadi model utama sejak peluncuran Windows Phone 7. Kotak yang berlabel "CoreApplication" merupakan model aplikasi baru untuk Windows Phone yang merupakan bagian dari model aplikasi Windows 8. Saat Windows Phone 8 rilis, model aplikasi ini hanya murni mendukung *native* aplikasi yang menggunakan Direct3D [10].

Kedua model aplikasi yang bergantung pada sebuah *shared set* dari layanan *core platform*. Sebgaian besar dari aplikasi *store* hanya pernah menampilkan layanan ini secara tidak langsung.

II.4. Speech Recognition

Speech recognition merupakan salah satu jenis *biometric recognition*, yaitu proses komputer untuk mengenali setiap yang diucapkan seseorang berdasarkan perintah suara yang dikonversi kedalam *digital print* [3]. Perancangan sistem untuk pengolahan suara terdiri dari fase-fase berikut:

1. Fase pertama, fase ini merupakan fase pembelajaran atau dikenal juga sebagai *training phase*. Berikut ini adalah diagram dari fase pembelajaran
2. Fase kedua, fase ini merupakan fase pengujian atau *operating session*.

II.5. Bing Speech Services

Bing Speech Services adalah teknologi dari Microsoft Corporation yang menyediakan layanan untuk memproses bahasa natural dalam hal ini adalah bahasa manusia. Layanan ini dapat diintegrasikan ke dalam aplikasi yang dibuat oleh pengembang, dalam hal ini aplikasi yang cocok adalah aplikasi dengan platform Windows dan Windows Phone.

Adapun layanan yang disediakan oleh *Bing Speech Services* adalah sebagai berikut [4]:

1. *Voice Commands* adalah layanan yang memungkinkan pengguna untuk mengoperasikan aplikasi dengan suara mereka

2. *Speech Recognition* adalah layanan yang memungkinkan aplikasi menerima pesan ucapan pengguna sebagai masukan yang kemudian akan dikonversi ke dalam bentuk teks.

Speech Synthesis adalah layanan yang memungkinkan aplikasi dapat membunyikan teks ke dalam bentuk suara.

II.6. Object Relational Mapping

Object Relational Mapping (ORM) adalah suatu teknik pemrograman yang digunakan untuk menghubungkan kode objek dengan *database* relasional. Kode objek ditulis dalam bahasa berorientasi objek pemrograman (OOP) seperti Java atau C#. Konsep ORM adalah melakukan pemetaan dari tabel menjadi objek. Kolom-kolom yang ada pada tabel nantinya akan menjadi variabel-variabel dalam objek tersebut. Satu objek mewakili satu baris. Karena merupakan objek, untuk mengakses beberapa baris sekaligus dapat disamakan dengan mengakses array dari objek tersebut. ORM memiliki kemampuan untuk menciptakan objek *database virtual*, yaitu suatu model *database* yang direpresentasikan ke dalam sebuah objek pada bahasa pemrograman OOP. Adapun kelebihan yang dimiliki oleh ORM adalah mempercepat pengembangan program. Contohnya yaitu mengurangi perulangan kode *query*, dan memudahkan pemakaian karena tabel yang ada sudah tersusun dalam bentuk objek. Singkatnya ORM ini adalah sebuah konsep untuk menggabungkan metode pemrograman *database* dengan konsep *Object Oriented Programming* (OOP) [11].

II.7. JSON (Javascript Object Notation)

JSON (JavaScript Object Nation) adalah format pertukaran data ringan, mudah dibaca dan ditulis oleh manusi a, serta mudah diterjemahkan dan dibuat (generate) oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemograman JavaScript, Standar ECMA-262 Edisi ke-3-Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemograman apapuun

karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Phyton dll. Oleh karena sifat-sifat tersebut , menjadikan JSON ideal sebagai bahasa pertukaran data [12].

II.8. API (Application Programming Interface)

Application programming interface (API) adalah sekumpulan perintah, fungsi, komponen, dan protokol yang disediakan oleh sistem operasi ataupun bahasa pemograman tertentu yang dapat digunakan oleh programmer saat membangun perangkat lunak.

Dalam suatu pemograman dibutuhkan setidaknya ribuan *system call* per detik oleh karena itu banyak programmer yang menggunakan API. Didalam API terdapat fungsi-fungsi/perintah untuk menggantikan bahasa yang digunakan dalam s*ystem calls* dengan bahasa yang lebih mudah dimengerti oleh programmer. Fungsi yang dibuat dengan menggunakan API tersebut kemudian akan memanggil *system call* sesuai dengan sistem operasinya. Tidak tertutup kemungkinan nama dari *system call* sama dengan nama API [13].

Kelebihan pemograman menggunakan API adalah:

1. Portabilitas

   Programmer yang menggunakan Api dapat menjalankan programnya dengan sistem operasi mana saja asalkan sudah terinstal API tersebut [13].
2. Mudah Dimengerti

   API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call* [13].

II.9. Object Oriented Analysis and Design

Analisis dan Desain Berorientasi Objek (*Object Oriented Analysis and Design*) adalah cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep. Dasar pembuatannya sendiri adalah objek yang merupakan kombinasi antara struktur data dan perilaku dalam satu entitas. Alasan

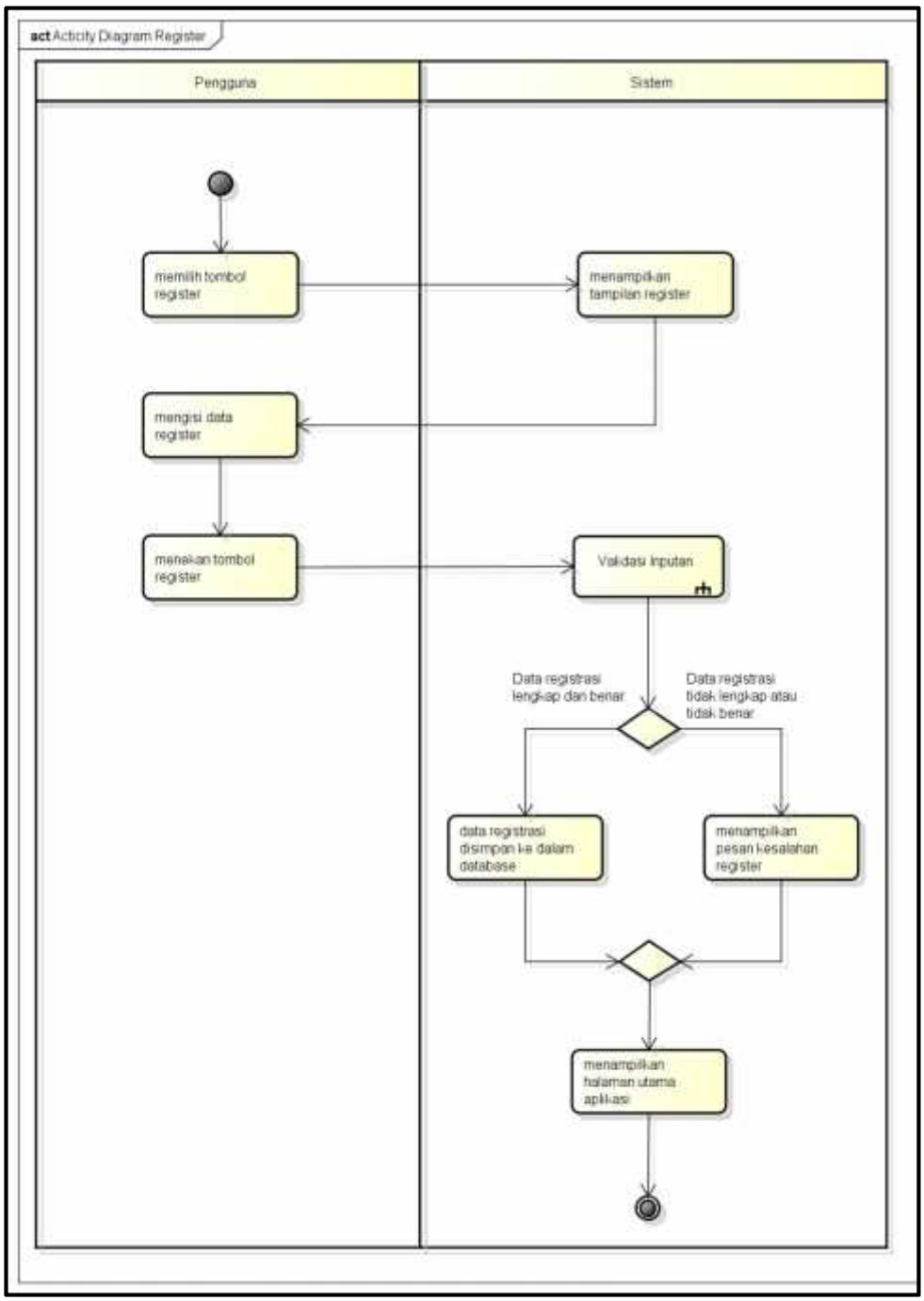
mengapa harus memakai metode beorientasi objek yaitu karena perangkat lunak itu sendiri yang bersifat dinamis, di mana hal ini disebabkan karena kebutuhan pengguna berubah dengan cepat. Selain itu bertujuan untuk menghilangkan kompleksitas transisi antar tahap pada pengembangan perangkat lunak, karena pada pendekatan berorientasi objek, notasi yang digunakan pada tahap analisis perancangan dan implementasi relatif sama tidak seperti pendekatan konvensional yang dikarenakan notasi yang digunakan pada tahap analisisnya berbeda-beda hal itu menyebabkan transisi antar tahap pengembangan menjadi kompleks. Di samping itu dengan pendekatan berorientasi objek membawa pengguna kepada abstraksi atau istilah yang lebih dekat dengan dunia nyata, karena di dunia nyata itu sendiri yang sering pengguna lihat adalah objeknya bukan fungsinya. Beda ceritanya dengan pendekatan terstruktur yang hanya mendukung abstraksi pada level fungsional. Adapun dalam pemrograman berorientasi objek menekankan berbagai konsep seperti: *Class, Object, Abstract, Encapsulation, Polymorphism, Inheritance* dan tentunya UML (*Unified Modeling Language*). UML (*Unified Modeling Language*) sendiri merupakan salah satu alat bantu yang dapat digunakan dalam bahasa pemrograman berorientasi objek. Selain itu UML merupakan *standard modeling language* yang terdiri dari kumpulan-kumpulan diagram, dikembangkan untuk membantu para pengembang sistem (*developer*) dan software agar bisa menyelesaikan tugas-tugas seperti: Spesifikasi, Visualisasi, Desain Arsitektur, Konstruksi, Simulasi dan Testing. Dapat disimpulkan bahwa UML (*Unified Modeling Language*) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk memvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis Objek (*Object Oriented Programming*) [14].

Dokumentasi UML menyediakan 10 macam diagram untuk memodelkan aplikasi berorientasi objek yang 4 diantaranya adalah [15]:

1. *Activity Diagram*

Activity Diagram adalah sebuah tahapan yang lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Di mana

biasanya dipakai pada *business modeling* untuk memperlihatkan urutan aktifitas proses bisnis. *Activity Diagram* ini sendiri memiliki struktur diagram yang mirip flowchart atau data flow diagram pada perancangan terstruktur. *Activity Diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case diagram*. Contoh activity diagram dapat dilihat pada Gambar II.2 berikut.



Gambar II.2 Contoh *Activity Diagram Register* ke dalam sistem

2. *Use Case Diagram*

Use Case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Di dalam *use case diagram* ini sendiri lebih ditekankan kepada apa yang diperbuat sistem dan bagaimana sebuah sistem itu bekerja. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan bentuk dari sebuah pekerjaan tertentu, misalnya *login* ke dalam sistem, *posting*, dan sebagainya, sedangkan seorang aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [16]. Adapun komponen-komponen dalam use case diagram diantaranya:

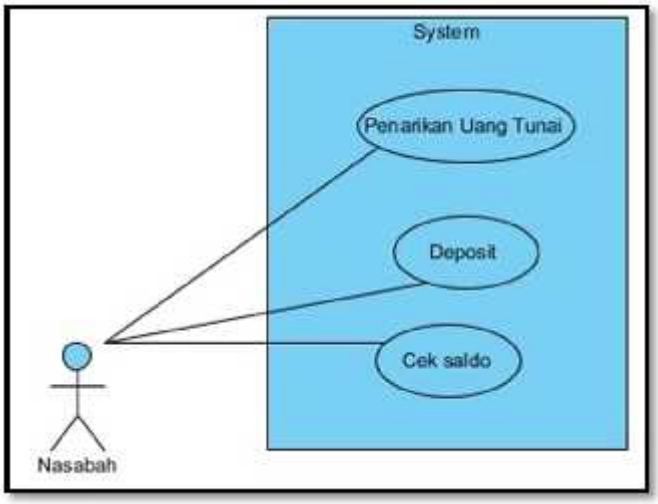
a. Aktor

Aktor merupakan suatu entitas yang berkaitan dengan sistem tapi bukan dari bagian dalam sistem itu sendiri. Aktor berada diluar sistem namun berkaitan erat dengan fungsionalitas didalamnya. Aktor dapat memiliki hubungan secara langsung terhadap fungsi utama baik terhadap salah satu atau semua fungsionalitas utama. Aktor juga dapat dibagi terhadap berbagai jenis atau tingkatan dengan cara digeneralisasi atau dispesifikasi tergantung kebutuhan sistemnya. Aktor biasanya dapat berupa pengguna atau database yang secara pandang berada dalam suatu ruang lingkup sistem tersebut.

b. Use Case

Use case merupakan gambaran umum dari fungsi atau proses utama yang menggambarkan tentang salah satu perilaku sistem. Perilaku sistem ini terdefinisi dari proses bisnis sistem yang akan dimodelkan. Tidak semua proses bisnis digambarkan secara fungsional pada use case, tetapi yang digambarkan hanya fungsionalitas utama yang berkaitan dengan sistem. Use case menitik beratkan bagaimana suatu sistem dapat berinteraksi baik antar sistem maupun diluar sistem.

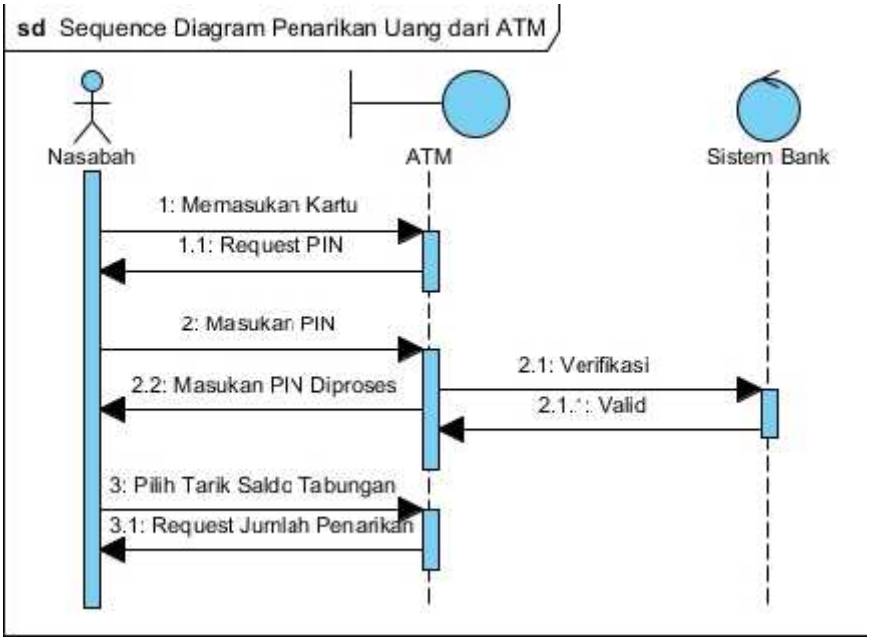
Contoh *use case diagram* dapat dilihat pada Gambar II.3 berikut.



Gambar II.3 Contoh *Use Case Diagram* Aktivitas Nasabah di ATM

3. *Sequence Diagram*

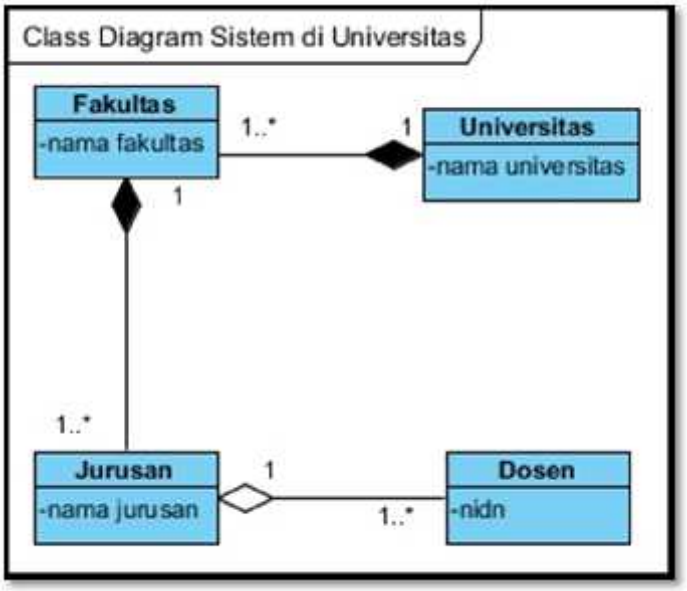
Sequence Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram jenis ini memberikan kejelasan sejumlah objek dan pesan-pesan yang diletakkan diantaranya di dalam sebuah *use case*. Komponen utamanya adalah objek yang digambarkan dengan kotak segi empat atau bulat, *message* yang digambarkan dengan garis penuh, dan waktu yang ditunjukkan dengan *progress vertical*. Manfaat dari *sequence diagram* adalah memberikan gambaran detail dari setiap *use case diagram* yang dibuat sebelumnya [16]. Contoh *sequence diagram* dapat dilihat pada Gambar II.4 berikut.



Gambar II.4 Contoh *Sequence Diagram* Penarikan Uang dari ATM

4. *Class Diagram*

Class Diagram adalah sebuah *class* yang menggambarkan struktur dan penjelasan *class*, paket, dan objek serta hubungan satu sama lain. *Class* diagram juga menjelaskan hubungan antar *class* secara keseluruhan di dalam sebuah sistem yang sedang dibuat dan bagaimana caranya agar mereka saling berkolaborasi untuk mencapai sebuah tujuan [16]. Contoh *class diagram* dapat dilihat pada Gambar II.5 berikut.



Gambar II.5 Contoh *Class Diagram* Sistem di Universitas

II.10. Skala Likert

Skala *likert* merupakan salah satu metode yang dapat digunakan untuk mendapatkan hasil pengujian angket atau kuesioner. Skala *likert* sering digunakan untuk mengukur sikap, pendapat, dan presepsi seseorang atau sekelompok orang tentang suatu gejala atau fenomena. Skala *likert* meminta kepada responden untuk menjawab suatu pernyataan dengan jawaban sangat setuju (SS), setuju (S), belum memutuskan (N), tidak setuju (T), dan sangat tidak setuju (ST). Untuk masing-masing jawaban mempunyai bobot nilai yang berbeda, adapun pemberian bobot nilai itu sendiri tergantung jenis pernyataan yang dapat dilihat pada Tabel II.1 berikut [17]

Tabel II.1 Pembobotan Nilai Skala Likert

Jenis Pertanyaan	Bobot Pendapat				
	SS	S	N	T	ST
Positif	5	4	3	2	1
Negatif	1	2	3	4	5

Ketika jawaban sudah dihitung berdasarkan bobot nilainya masing-masing, maka dilanjutkan menghitung rata-rata dari setiap konten pernyataan dan indikator pernyataan itu sendiri dengan menggunakan rumus:

$$x = \sum_{i=1}^{n} x \, / n$$

Dimana n adalah jumlah responden, sedangkan x adalah jumlah konten nilai ataupun konten pernyataan. Sedangkan untuk menghitung rata-rata dari indikator, n akan berupa jumlah konten pernyataan, sedangkan x adalah jumlah dari konten indikator pernyataan [17].