BAB II
TINJAUAN PUSTAKA DAN LANDASAN TEORI

2.1 Tinjauan Pustaka

Pembuatan dan penggunaan fitur admin pada suatu sistem seperti ini telah banyak dilakukan sebelumnya dalam sektor pendidikan, yang secara umum memiliki tujuan yang sama yaitu mengatur agar penggunaan sistem dapat berjalan dengan lancar, seperti pengaturan otorisasi *user* serta memasukkan dan memperbaharui data yang terdapat pada sistem.

Bobby Melky Tulangow (2011) dalam "*Sistem Ujian Berbasis Web*", membuat web yang dapat melayani pelaksanaan ujian secara *online*, sehingga proses ujian yang dahulu terlalu rumit bisa disederhanakan. Pada sistem ini terdapat *Sistem Administrator* yang bertanggung jawab atas keamanan sistem dengan mengatur otorisasi *user*. Pengaturan otorisasi termasuk didalamnya adalah pembuatan fitur *login* untuk membatasi hak akses *user*, menambah, memperbaharui dan menghapus *user* dan membatasi fungsi yang dapat dilakukan oleh *user* sesuai dengan perannya.

Anggiani Septima Riyadi, Eko Retnandi, dan Asep Deddy (2012) telah melakukan penelitian dengan judul "Perancangan Sistem Informasi Berbasis *Website* Subsistem Guru di Sekolah Pesantren Persatuan Islam 99 Rancabango". Sistem ini bertujuan untuk memberikan kemudahan dalam penyampaian informasi terkait sekolah tersebut, dan kemudahan dalam aktivitas-aktivitas akademik. Pada sistem tersebut terdapat admin yang bertugas untuk mengelola data yang terdapat pada sistem, seperti melakukan hapus, tambah dan edit pada data guru, menambah dan menghapus materi ajar, serta memperbaharui kalender akademik.

.Tedi Kurnia, Dini Destiani, dan Asep Deddy Supriatna (2012) telah melakukan penelitian dengan judul "Perancangan Sistem Informasi Akademik Nilai Siswa Berbasis Web " dengan studi kasus SMK Ciledug Al-Musaddadiyah Garut. Sistem ini bertujuan untuk memberikan kemudahan bagi guru untuk dapat mengelola data-data nilai harian siswa, dan siswa dapat menerima informasi dengan cepat karena dapat diakses lewat internet. Pada sistem ini terdapat admin

yang bertugas memberikan layanan seperti memasukkan data-data nilai siswa, mencetak laporan, dan sebagainya.

Berdasarkan contoh aplikasi tersebut, dapat dilihat banyaknya tugas yang harus dikerjakan oleh seorang admin, baik terkait pengaturan otorisasi dan otentikasi maupun terkait pengolahan data yang terdapat pada sistem, sehingga dapat dilihat bahwa fitur admin merupakan fitur yang penting dalam pembuatan suatu sistem. Oleh karena itu, pada penelitian ini fitur admin juga perlu ditambahkan.

Dalam penelitian ini, selain berperan penting dalam pengaturan hak akses dan pengolahan data pada sistem, admin juga dituntut untuk mengolah *tag* yang akan digunakan untuk pembuatan soal, pengolahan yang dimaksud termasuk di dalamnya penambahan *tag*, perbaharuan *tag* jika dibutuhkan, dan penghapusan *tag* jika sudah tidak digunakan.

2.2 Landasan Teori

2.2.1 Sistem Administrator

Sistem Administrator adalah seseorang yang bekerja untuk memelihara dan mengoperasikan sebuah sistem komputer atau jaringan yang berjalan setiap harinya pada suatu organisasi/perusahaan. Menurut Nugraha Pengestu dan Ryan R.Adhisa (2013) Sistem Administrator (dikenal juga sebagai admin, administrator, sysadmin, site admin, dll) merupakan profesi yang memiliki tugas untuk melakukan administrasi terhadap sistem, pemeliharaan sistem, memiliki kewenangan mengatur hak akses terhadap sistem, serta hal – hal lain yang berhubungan dengan pengaturan operasional sebuah sistem.

Tugas utama seorang sistem administrator adalah memastikan sistem tetap berjalan lancar dalam memberikan pelayanan kepada penggunanya.

2.2.2 Asesmen dalam Pembelajaran

Asesmen merupakan segala tindakan yang dilakukan oleh guru maupun siswa untuk menilai pencapaian mereka dan menggunakan hasilnya untuk mengubah aktivitas belajar mengajar mereka (Black, et al., 2001).

Untuk menjalankan asesmen yang mampu meningkatkan pembelajaran dengan baik, ada beberapa hal yang harus dipenuhi (Black, et al., 2001):

1. Asesmen harus menghasilkan informasi yang akurat.
2. *Feedback* yang diberikan kepada siswa lebih baik deskriptif dibanding evaluatif.
3. Siswa diikutsertakan dalam proses asesmen.

Dalam mengikutsertakan siswa dalam proses asesmen, Stiggins et al. (2007) menjelaskan bahwa siswa perlu untuk mengetahui apa yang seharusnya mereka tahu, apa yang mereka tahu sekarang, dan bagaimana cara memperkecil jarak antara keduanya. Stiggins et al. (2007) kemudian menjelaskan bagaimana sebaiknya asesmen dilakukan untuk membantu siswa dalam mengetahui jawaban atas 3 hal tersebut:

1. Memberikan visi dari target pembelajaran dengan jelas dan mudah dipahami.
2. Memberikan contoh dan *model* mengenai pekerjaan atau hasil yang baik dan buruk.
3. Memberikan *feedback* yang deskriptif secara regular.
4. Mengajarkan siswa untuk menetapkan tujuan dan melakukan asesmen pribadi.
5. Mendesain pelajaran agar fokus pada satu aspek dalam satu waktu.
6. Mengajarkan siswa bagaimana membuat revisi pembelajaran yang terfokus.
7. Mengajak siswa untuk melakukan *tracking*, refleksi, dan komunikasi mengenai perkembangan pembelajaran mereka secara pribadi.

2.2.3 *Tag*

Tag atau yang dalam Bahasa Indonesia dikenal dengan label menurut Marinus Angipora (2002) merupakan suatu bagian dari sebuah produk yang membawa informasi verbal tentang produk atau penjualnya.

Menurut Tjiptono (1997) label merupakan bagian dari suatu produk yang menyampaikan informasi mengenai produk dan penjual. Sebuah label biasa

merupakan bagian dari kemasan, atau bisa pula merupakan etiket (tanda pengenal) yang dicantelkan pada produk.

Tag atau label digunakan untuk mengidentifikasi suatu produk, sehingga produk dapat dikelompokkan dan dibedakan dengan produk lain berdasarkan label yang terdapat pada produk.

2.2.4 UML

Unified Modelling Language (UML) adalah alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual (Braun, et. al. 2001). Selain itu merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Whitten, et. al. 2004).

Penggunaan UML bertujuan untuk mengidentifikasikan bagian-bagian yang termasuk dalam lingkup sistem didalam aplikasi, mendokumentasikan hasil analisa dan desain serta untuk menggambarkan sebuah sistem *software*. *Model* UML yang dipakai dalam pengembangan aplikasi ini antara lain *Use Case* Diagram dan *Activity* diagram.

2.2.5 Diagram – Diagram UML

Menurut situs resminya, diagram UML dapat dibedakan atas *Structure diagram* dan *Behavior diagram*. *Structure diagram* adalah diagram yang menampilkan susunan statis sistem dan bagian – bagian pada implementasi dan menunjukkan bagaimana bagian – bagian itu saling terkait satu sama lain. Sedangkan *Behavior diagram* menunjukan perilaku dinamis objek dalam suatu sistem yang dapat digambarkan sebagai rangkaian perubahan sistem dari waktu ke waktu.

1. ***Use Case Diagram***

   Use Case Diagram merupakan salah satu *Behavior diagram* yang terdiri dari satu set fungsi (*use case*) yang terdapat pada sistem dan dapat dilakukan oleh satu atau lebih *user* sistem (aktor) untuk memberikan hasil yang dapat diamati. *Use Case Diagram* mendeskripsikan interaksi antara *user* (aktor) dengan sistem yang dibuat.

2. *Activity Diagram*

Activity diagram menurut Martin Fowler (2005 : 163) adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. Node pada sebuah *Activity diagram* disebut sebagai *action*, sehingga *Activity diagram* menampilkan sebuah *activity* yang tersusun dari *action*. *Activity diagram* Menunjukan kegiatan dan perubahan dari satu aktivitas ke aktivitas lainnya dengan peristiwa yang terjadi di beberapa bagian dari sistem.

3. *Class Diagram*

Class Diagram merupakan salah satu *Structure diagram* yang mengambarkan struktur sistem yang dirancang. Subsistem atau komponen digunakan sebagai kelas dan antarmuka terkait. *Class* diagram menurut Munawar (2005 : 28) merupakan himpunan dari objek-objek yang sejenis. Sebuah objek memiliki keadaan sesaat (*state*) dan perilaku (*behavior*). *State* sebuah objek adalah kondisi objek tersebut yang dinyatakan dalam *attribute/properties*. Sedangkan perilaku suatu objek mendefinisikan bagaimana sebuah objek bertindak/beraksi dan memberikan reaksi.

Class Diagram menggambarkan relasi atau hubungan antar *class* dari sebuah sistem. Berikut ini beberapa gambaran relasi yang ada dalam *Class Diagram*.

a. Asosiasi

Asosiasi merupakan hubungan antar *class* yang statis. *Class* yang mempunyai relasi asosiasi menggunakan *class* lain sebagai atribut pada dirinya. Asosiasi digambarkan dengan garis panah lurus.

b. Agregasi

Agrerasi merupakan relasi yang membuat *class* yang saling berelasi terikat satu sama lain namun tidak terlalu bergantung.

c. Komposisi

Komposisi merupakan relasi agregasi dengan mengikat satu sama lain dengan ikatan yang sangat kuat dan saling bergantung satu sama lain.

d. *Dependency*

Merupakan hubungan antar-*class* di mana *class* yang memiliki relasi *dependency* menggunakan *class* lain sebagai *attribute* pada *method.*

2.2.6 *Entity Relationship Diagram*

Menurut Sutanta (2011:91), "*Entity Relationship Diagram* (ERD) merupakan suatu *model* data yang dikembangkan berdasarkan objek." *Entity Relationship Diagram* (ERD) digunakan untuk menjelaskan hubungan antar data dalam basis data kepada *user* secara logis.

Bagi perancang atau analis sistem, *Entity Relationship Diagram* (ERD) berguna untuk memodelkan sistem basis data yang nantinya akan dikembangkan. *Model* ini juga membantu perancang atau analis sistem pada saat melakukan analisis dan perancangan basis data karena *model* ini dapat menunjukkan macam data yang dibutuhkan dan kerelasian antar data didalamnya.

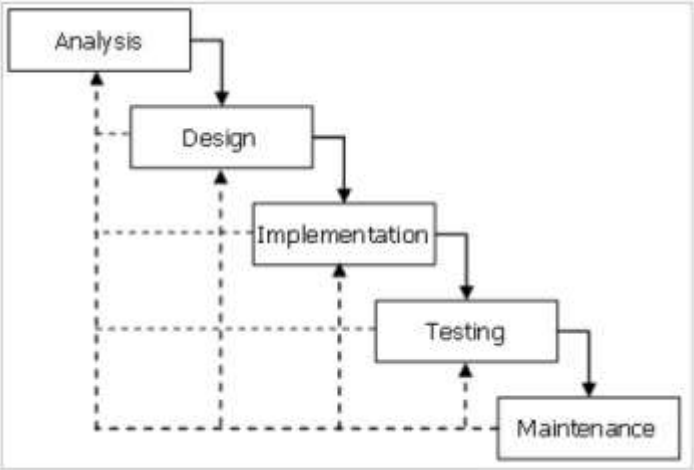
2.2.7 Metode *Waterfall* dalam *Software Development Life Cycle*

SDLC (*Software Development Life Cycle*) merupakan sebuah siklus pengembangan perangkat lunak yang terdiri dari beberapa tahapan-tahapan penting dalam membangun perangkat lunak yang dilihat dari segi pengembangannya. Tahapan-tahapan tersebut diantaranya : perencanaan (*planning*), analisis (*analysis*), desain (*design*), implementasi (*implementation*), dan uji coba (*testing*). Selain untuk proses pembuatan, SDLC juga penting untuk proses *maintenance* (pemeliharaan) *software.*

Salah satu metode yang terdapat pada *model* SDLC adalah metode *Waterfall. Model* ini Pertama kali diperkenalkan oleh Winston Royce pada tahun 1970. *Model* ini merupakan *model* klasik yang sederhana dengan aliran sistem yang linear, yang artinya suatu tahapan harus selesai terlebih dahulu, sebelum

memulai tahapan yang lain. *Output* dari setiap tahap menjadi *input* bagi tahap berikutnya.

Salah satu referensi yang terkenal mengenai metode *waterfall* adalah metode yang dikemukakan oleh Yousef Bassil. Menurut Bassil (2011)," *Model Waterfall* SDLC adalah proses pengembangan perangkat lunak yang berurutan (*sequential*) dimana prosesnya dari atas ke bawah (seperti air terjun) melalui tahapan-tahapan yang harus dijalankan untuk keberhasilan pembuatan perangkat lunak". Untuk lebih memahami metode *waterfall* menurut Bassil, dapat dilihat pada **Gambar 2.1**.



Gambar 2.1 Metode *Waterfall* Menurut Referensi Bassil

(Sumber: Jurnal "A Simulation *Model* for the *Waterfall* Software Development Life Cycle", Youssef Bassil, 2011)

2.2.8 Visual Studio

Menurut situs resminya, Visual Studio adalah satu set lengkap alat pengembangan yang digunakan untuk membangun aplikasi Web ASP.NET, XML *Web Services*, aplikasi *desktop*, dan aplikasi *mobile*. Visual Basic, Visual C #, dan Visual C ++ menggunakan *Integrated Development Environment* (IDE) yang sama, yang memungkinkan alat untuk berbagi dan memudahkan terciptanya solusi dalam menggunakan bahasa campuran. Selain itu, bahasa-bahasa ini menggunakan fungsionalitas .NET *Framework*, yang menyediakan akses ke teknologi kunci yang mempermudah pengembangan aplikasi Web ASP dan XML *Web Services.*

2.2.9 Metode MVC dan ASP.NET

Aplikasi web yang dijadikan penelitian adalah aplikasi web yang dibangun menggunakan metode MVC (*Model-View-Vontroler*). *Model-View-Controller* atau MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (*Model*) dari tampilan (*View*) dan bagaimana cara memprosesnya (*Controller*). Dalam implementasinya, kebanyakan *framework* dalam aplikasi web adalah berbasis arsitektur MVC (*Model-View-Controller*) (Rosmala, Ichwan, & Muhammad, 2011).

Aplikasi web akan dibangun menggunakan ASP.NET (*Active Server Pages. NET*) yang mendukung metode MVC. ASP.NET merupakan platform pembuatan aplikasi web yang menyatu dengan .NET *Framework* serta menyediakan fasilitas-fasilitas bagi developer untuk membangun aplikasi web untuk level perusahaan.

2.2.10 MS SQL *Server*

Menurut (Agus, 2013:11), MS SQL *Server* adalah salah satu produk *Relational Database Management System* (RDBMS) populer yang berfungsi sebagai relasi *database* dalam sebuah program aplikasi. Microsof SQL *Server* mendukung SQL sebagai bahasa pemograman *querry*. SQL yang sebagimana sudah diketahui secara luas merupakan bahasa standar internasional untuk proses *querry database*.

2.2.11 *Black box Testing*

Black box testing adalah pengujian yang dilakukan hanya dengan mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. *Black box testing* dianalogikan seperti kita melihat suatu kotak hitam, kita hanya bisa melihat penampilan luarnya saja, tanpa tau ada apa dibalik bungkus hitam nya. Sama seperti pengujian *black box*, mengevaluasi hanya dari tampilan luarnya (*interface* nya) dan fungsionalitasnya, tanpa mengetahui apa sesungguhnya yang terjadi dalam proses detilnya (hanya mengetahui input dan output).