BAB II

LANDASAN TEORI

2.1 *Test Of English as a Foreign Language (TOEFL)*

TOEFL adalah bentuk tes khusus bahasa Inggris standart sebagai bahasa asing yang ditujukan kepada mereka yang bukan *native speaker* (Rudman 2011). Sedangkan tujuan utama TOEFL ini adalah untuk mengukur sejauh mana kemampuan bahasa Inggris seseorang menurut ukuran standar yang telah ditetapkan.

Saat ini tes ini sangat dibutuhkan terutama oleh universitas. Selain itu, institusi seperti pemerintahan, bisnis, atau program beasiswa juga membutuhkan tes ini. Tes TOEFL pada umumnya terdiri dari 2 bentuk, yaitu : *paper based* dan *Computer Based.*

2.2 *Manfaat TOEFL*

Menurut Rudman (2011) Sertifikat TOEFL pada masa ini sangat dibutuhkan terutama untuk syarat penerimaan pada perusahaan maupun universitas yang menggunakan standart bahasa Inggris, sebagai berikut :

a. Syarat melakukan pendaftaran program *short course* dan *non-degree* program di negara-negara berbahasa Inggris.
b. Syarat pendaftaran dan penempatan dalam program kolaborasi internasional yang menggunakan bahasa Inggris sebagai bahasa pengantarnya.
c. Sebagai syarat mendaftar program beasiswa ke berbagai negara (sebagai seleksi awal).

d. Sebagai syarat bagi para pelamar kerja maupun kenaikan jabatan atau promosi.
e. Sebagai monitor perkembangan bahasa Inggris khususnya yang memerlukan kecakapan/keahlian dalam bahasa Inggris.

2.3 *Jenis-Jenis TOEFL*

Menurut Sharpe (2012) dalam bukunya *Barron's How To Prepare For the TOEFL* terdapat empat jenis TOEFL, yaitu :

a. Paper and Pencil TOEFL
b. Computer Based TOEFL
c. Institutional TOEL
d. International TOEFL

Paper and Pencil TOEFL dan *Computer Based TOEFL* sering dikategorikan sebagai *Official Administrations*, biasanya dilaksanakan setiap bulan. Sedangkan *Institutional TOEFL* dilakukan sebagai prasyarat pada penerimaan mahasiswa baru S2 dan S3, penerimaan pegawai baru yang biasanya dilakukan diawal tahun atau bersifat *incidental*. Sedangkan International TOEFL adalah TOEFL yang diakui berbagai negara di dunia.

2.4 *Aspek-Aspek TOEFL*

Menurut Muhammad (2007); Materi yang diujikan di dalam TOEFL meliputi penguasaan terhadap empat keterampilan berbahasa Inggris. Secara umum materi tersebut dapat dikelompokkan kedalam tiga bagian, yaitu:

1. Bagian *Listening Comprehension*
   a. Short Conversation

b. *Longer Conversation*

c. *Lectures and Taks*

2. Bagian *Structure and Written Expression*

a. *Sentence Completion*

b. *Error Identification*

3. Bagian *Reading Comprehension and Vocabulary*

a. *Science*

b. *North American History, Goverment of Geography*

c. *Art and Literature, Biographies of Famous People*

Selain ketiga bagian di atas ada bagian TOEFL yang biasa disebut dengan TWE (*Test of Written English*). TWE adalah bentuk tes tertulis berbentuk esai yang biasanya diberikan pada tiga puluh menit sebelum pelaksanaan tes TOEFL. Namun tidak jarang pula, tes ini dilakukan ketika akhir pelaksanaan tes TOEFL. TWE berbentuk esai dgan batasan 200 – 300 kata.

2.5 *Scoring* TOEFL

Menurut Longman (2003) scoring TOEFL adalah kalkulasi dari semua tahap pekerjaan . Skor ini didasarkan pada jumlah jawaban yang benar dari setiap peserta ujian. Skor untuk setiap bagian dihitung dengan dua skala statistik. Skor setiap TOEFL umumnya memiliki skala 31-68, skor total memiliki jangkauan 310-667.

Cara menghitung skor pada tes TOEFL adalah sebagai berikut :

$$\frac{(\text{jumlah benar ketiga bagian}) \times 10}{3}$$

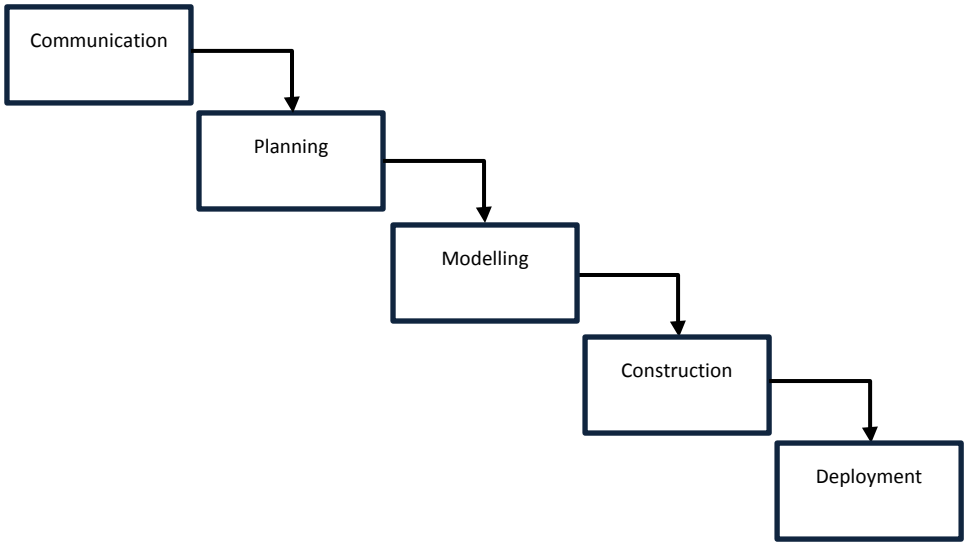
2.6 *Aplikasi*

Menurut Noviansyah (2008), aplikasi adalah penggunaan atau penerapan suatu konsep yang menjadi suatu pokok pembahasan. Aplikasi dapat diartikan juga sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu. Aplikasi software yang dirancang untuk suatu tugas khusus dapat dibedakan menjadi dua jenis, yaitu:

a. Aplikasi *software* spesialis, program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.
b. Aplikasi *software* paket, suatu prrgram dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu.

2.7 *Software Development Life Cycle*

Menurut Pressman (2010) di dalam *Software Development Life Cycle* (SDLC) terdapat beberapa model. Salah satu model dalam SDLC adalah model *waterfall*, terkadang disebut sebagai siklus hidup klasik. *Waterfall* dilakukan untuk penyebaran perangkat lunak yang dimulai dengan spesifikasi permintaan pelanggan dan berlangsung melalui perencanaan, pemodelan, *construction* dan *deployment* yang berakhir pada dukungan yang berkelanjutan dari terselesainya *software*. Gambar waterfall dapat dilihat pada Gambar 2.1



Gambar 2.1 SDLC dengan metode *Waterfall* (Pressman, 2010)

1. *Communication* (komunikasi)

   Langkah ini merupakan analisis terhadap kebutuhan *software*, dan tahap untuk mengadakan pengumpulan data dengan melakukan pertemuan dengan *customer*, maupun mengumpulkan data-data tambahan baik yang ada di jurnal, artikel, maupun dari internet.

2. *Planning* (perencanaan)

   Proses *planning* merupakan lanjutan dari proses *communication (analysis requirement)*. Tahapan ini menggambarkan tugas-tugas teknis yang dilakukan, sumber daya yang dibutuhkan, produk yang harus dihasilkan, dan jadwal-jadwal kerja termasuk rencana yang akan dilakukan.

3. *Modeling* (pemodelan)

   Proses *modeling* ini akan menerjemahkan syarat kebutuhan-kebutuhan menjadi sebuah perancangan *software* yang dapat diperkirakan sebelum

dibuat *coding*. Proses ini berfokus pada rancangan struktural data, arsitektur *software*, representasi *interface*, dan detail (algoritma) prosedural.

4. *Construction* (konstruksi)

   Construction merupakan proses membuat kode. *Coding* atau pengkodean merupakan penerjemahan desain dalam bahasa yang bisa dikenali oleh komputer. *Programmer* akan menerjemahkan transaksi yang diminta oleh *user*. Tahapan inilah yang merupakan tahapan secara nyata dalam mengerjakan suatu *software*, artinya penggunaan komputer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap perangkat lunak yang telah dibuat tadi. Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap perangkat lunak tersebut untuk kemudian bisa diperbaiki.

5. *Deployment* (pengoperasian)

   Tahapan ini bisa dikatakan akhir dalam pembuatan sebuah *software* atau sistem. Setelah melakukan analisis, desain dan pengkodean maka sistem perangkat lunak yang sudah jadi akan digunakan oleh *user*. Kemudian *software* yang telah dibuat harus dilakukan pemeliharaan secara berkala.

2.8 *Website*

Menurut Sutarman (2003), *website* (situs *web*) adalah merupakan alamat (URL) yang berfungsi sebagai tempat penyimpanan data dan informasi dengan berdasarkan topik tertentu. Sedangkan, *web page* (halaman web) merupakan halaman khusus dari situs web tertentu yang tersimpan dalam bentuk file. Di dalam halaman tersebut terdapat berbagai informasi dan *link* yang tersimpan serta

menghubungkan suatu informasi ke informasi lain dalam *page* sama ataupun web page lain pada *website* yang berbeda.

Pada situs/*web* dapat di kategorikan menjadi dua, yaitu *web* statis dan *web* dinamis. *Web* statis merupakan *web* yang berisi atau menampilkan informasi-informasi yang sifatnya tetap atau statis, sedangkan *web* dinamis merupakan *web* yang menampilkan informasi serta dapat berinteraksi dengan *user* yang sifatnya dinamis. Untuk membuat *web* dinamis dibutuhkan kemampuan pemrograman *web*. Terdapat 2 kategori dalam pemrograman *web* :

1. *Server - side programming*
2. *Client - side programming*

Pada *server-side programming*, perintah program yang dijalankan di *web* server, kemudian hasilnya dikirimkan ke *browser* dalam bentuk HTML biasa. Sedangkan *client – side programming* perintah program dijalankan di browser, sehingga ketika client meminta dokumen yang mengandung script, maka script tersebut akan di download dari servernya kemudian dijalankan di *browser* yang bersangkutan.

2.9 *Black Box Testing*

Menurut Rizky (2011), pengertian dari *Black Box Testing* adalah suatu tipe *testing* yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya. Berdasarkan hal tersebut, para *tester* memandang perangkat lunak seperti layaknya “kotak hitam” yang tidak terlihat isinya, tetapi mendapat proses *testing* bagian luarnya saja. *Black Box Testing* hanya memandang perangkat lunak dari sisi spesifikasi dan kebutuhan yang telah ditentukan pada awal perancangan. Keuntungan dari jenis *testing* ini antara lain:

1. Anggota tim *tester* tidak harus dari seseorang yang memiliki kemampuan teknis program.
2. Kesalahan dari perangkat lunak ataupun *bug* sering ditemukan oleh komponen *tester* yang berasal dari pengguna.
3. Hasil dari *black box testing* dapat memperjelas kontradiksi ataupun kerancuan yang mungkin timbul dari eksekusi sebuah perangkat lunak.
4. Proses *testing* dapat dilakukan lebih cepat dibandingkan *white box testing.*

Alasan penulis menggunakan *black box testing* dikarenakan dapat dengan mudah mengetahui fungsi aplikasi yang salah, dapat dengan mudah mengetahui kesalahan pada tampilan, dapat dengan mudah mengetahui kesalahan pada *database*, dan dapat dengan mudah mengetahui kesalah inisialisasi dan tujuan akhir.