

## BAB II

### TINJAUAN PUSTAKA

#### 2.1 Tinjauan Pustaka

Pada penelitian ini, ada beberapa referensi yang dijadikan landasan pendukung untuk pelaksanaan setiap langkah penelitian. Referensi yang digunakan merupakan penelitian terdahulu yang akan menjadi acuan dalam membangun perangkat lunak berupa aplikasi berbasis web. Pada Tabel 2.1 berikut dapat diperhatikan referensi yang digunakan pada penelitian ini.

Aplikasi yang dikembangkan pada penelitian ini akan dibuat berbasis *website* menggunakan metode siklus hidup pengembangan perangkat lunak (SDLC) *Test-Driven Development* (TDD) yang merupakan salah satu bentuk SDLC *Agile* seperti yang sudah dilakukan pada penelitian sebelumnya dengan judul *Comparative Study of Test-Driven Development (TDD), Behavior-Driven Development (BDD) and Acceptance Test-Driven Development (ATDD)* yang dilakukan oleh Myint Myint Moe [7]. Pada penelitian yang dilakukannya, didapatkan bahwa TDD digunakan saat pengembang melakukan pengujian, hal ini berlaku untuk memastikan seluruh fitur dan fungsi yang dikembangkan pada aplikasi ini terbebas dari error dan bug.

Kemudian pada penelitian yang dilakukan oleh Xinyu Chang dan Jing Li dengan judul *Improvement of excel data processing function based on Spring MVC Framework* dijelaskan bahwa penggunaan *JavaScript Object Notation* (JSON) untuk memproses data dari *excel* agar dapat diproses lebih cepat [16]. Hal ini akan digunakan pada penelitian tugas akhir kali ini pada aplikasi saat melakukan penilaian otomatis untuk jawaban secara massal dengan format berkas CSV.

Penelitian ini juga akan menggunakan teknologi *Express.js* karena memiliki performa yang cepat dibandingkan pesaingnya yaitu *Ktor* sebagaimana yang telah diteliti pada penelitian sebelumnya dengan judul *Express.js and Ktor web server performance A comparative study* yang dilakukan oleh Isac Glantz dan Hampus Hurtig pada tahun 2022 [17].

Untuk memeriksa jawaban secara otomatis, akan digunakan model pemrosesan bahasa alami yang sudah terlatih berupa model *Python* dalam bahasa Inggris dan Indonesia. Diperlukan penelitian pendukung yang dapat menjembatani antara bahasa *JavaScript* yang akan digunakan pada *website* dan *Python* yang akan menjalankan pemeriksaan jawaban, penelitian tersebut adalah TENSORFLOW.JS: MACHINE LEARNING FOR THE WEB AND BEYOND [18]. Dengan ini, aplikasi yang dikembangkan akan menggunakan *Node.js* sebagai teknologi penghubung *JavaScript* dan *Python*.

Model yang digunakan dapat membandingkan jawaban siswa terhadap guru dalam bahasa Inggris dan Indonesia menggunakan teknik sebagaimana dilakukan pada penelitian oleh Svanhvít Ingólfssdóttir tentang *lemmatization* untuk bahasa Islandia dengan judul Nefnir: A high accuracy lemmatizer for Icelandic [19].

Tabel 2.1 Perbandingan Refrensi

| No | Judul  | Permasalahan   | Metode  | Hasil   | Perbandingan  |
|----|--|--|---|---|---|
| 1  | TENSORFLOW.JS: MACHINE LEARNING FOR THE WEB AND BEYOND (2019)                      | <i>Library</i> pembelajaran mesin biasanya ditulis dalam bahasa <i>Python</i> atau <i>C++</i> . Namun, jumlah pengguna <i>JavaScript</i> untuk <i>frontend</i> dan <i>backend</i> semakin bertambah. Jurnal ini berisi panduan untuk menjembatani <i>Python</i> dengan <i>JavaScript</i> | -API ( <i>Application Programming Interface</i> ) | Menggunakan <i>Layers API</i> , didapatkan performa waktu sebesar<br>- 3426 ms dengan <i>JavaScript</i> biasa dengan 1x percepatan,<br>- 10 ms dengan <i>WebGL</i> dengan 342x percepatan,<br>- 3 ms dengan <i>Node.js</i> dengan 1105x percepatan. | Aplikasi yang akan dikembangkan akan menggunakan API dibangun dengan <i>Node.js</i> tanpa menggunakan <i>TensorFlow</i> . |
| 2  | Improvement of excel data processing function based on Spring MVC Framework (2022) | <i>Excel</i> yang sangat umum digunakan, masih dinilai kurang pada pemrosesan sekelompok data dan verifikasi data. Dengan berkembangnya bidang sains dan teknologi, diperlukan pemrosesan data berskala besar dan berkelanjutan.   | -JSON( <i>JavaScript Object Notation</i> )        | Berdasarkan <i>Spring MVC</i> + <i>EasyUI</i> dengan pengembangan <i>Java J2EE IDE</i> . Diciptakan <i>kit</i> pengembangan yang bisa menkonversi <i>excel</i> dan melakukan ekspor dan impor sekumpulan data dengan menkonversinya menjadi JSON.   | CSV akan dikonversi menjadi bentuk JSON menggunakan teknik <i>spawn JavaScript</i> .                                      |
| 3  | Nefnir: A high accuracy lemmatizer for Icelandic (2019)                            | Mencari bentuk morfologi dasar pada sebuah kumpulan tulisan dengan bahasa yang kaya akan morfologi.  | - <i>Nefnir</i><br>- <i>part-of-speech</i>        | <i>Nefnir</i> meraih akurasi sebesar 99.55% untuk teks yang ditandai dengan benar. <i>Part-of-speech</i> (PoS) mendapat akurasi sebesar 96.88% dengan teks yang ditandai dengan <i>PoS tagger</i> .   | Model yang digunakan pada penelitian ini dicapai menggunakan teknik <i>stemming</i> dan <i>lemmatization</i> .            |

| No | Judul  | Permasalahan  | Metode  | Hasil   | Perbandingan   |
|----|--|---|---|---|--|
| 4  | Express.js and Ktor web server performance A comparative study (2022)  | Membandingkan dua <i>framework</i> web berdasarkan waktu merespon untuk membantu <i>developer</i> memilih antara <i>Express.js</i> dan <i>Ktor</i>  | Melakukan test terhadap waktu respon menggunakan database melalui <i>Object Relational Mapper</i> (ORM) <i>Sequelize</i> untuk <i>Express.js</i> , dan <i>Exposed</i> untuk <i>Ktor</i> | <i>Express.js</i> memiliki waktu respon yang lebih baik (3 ms) secara keseluruhan daripada <i>Ktor</i> (106 ms). Namun penggunaan <i>Object Relational Mapper</i> pada <i>Ktor</i> lebih berpengaruh pada hasil daripada <i>Express.js</i> .      | Aplikasi yang akan dikembangkan pada penelitian ini akan dibangun menggunakan <i>Express.js</i> sebagai <i>framework</i> modul HTTP dari <i>Node.js</i> .                    |
| 5  | Comparative Study of Test-Driven Development (TDD), Behavior-Driven Development (BDD) and Acceptance Test-Driven Development (ATDD) (2019) | Membandingkan perbedaan antara <i>Test-Driven Development</i> (TDD), <i>Behavior-Driven Development</i> (BDD), dan <i>Acceptance Test-Driven Development</i> (ATDD) dalam pengembangan perangkat lunak dengan lingkungan pengembangan yang berbeda. | Studi banding metode pengembangan perangkat lunak.  | <p>TDD digunakan saat pengembang menulis dan menjalankan pengujian.</p> <p>BDD merincikan perilaku fitur menggunakan bahasa yang dapat dimengerti semua orang yang terkait dalam pembangunan.</p> <p>ATDD membuat implementasi lebih efektif.</p> | SDLC TDD akan digunakan pada penelitian ini untuk mengembangkan aplikasi penilaian esai otomatis guna meminimalisir kemungkinan adanya <i>bug</i> pada tahap penulisan kode. |

Pada Tabel 2.1 sebelumnya, dapat diperhatikan beberapa tinjauan pustaka yang menjadi landasan pada penelitian ini lengkap dengan permasalahan, metode, hasil, dan perbedaan penelitian penulis dengan landasan teori penelitian sebelumnya.

## 2.2 Dasar Teori

Ada beberapa teori yang akan digunakan untuk mendukung penelitian tugas akhir ini. Teori yang digunakan akan menjadi landasan, dan memperkuat pemahaman selama penelitian ini dilakukan. Berikut adalah teori yang digunakan pada penelitian tugas akhir ini:

### 2.2.1 Test-Driven Development

*Test-Driven Development* (TDD) merupakan metode pengembangan perangkat lunak *agile* yang dikenalkan oleh *Extreme Programming* (XP). TDD menggunakan pendekatan pengujian unit yang terfokus pada pengembang [7]. TDD bekerja dengan cara menuliskan kode pengujian dan fungsionalitas hingga pengujian berhasil, kemudian menuliskan kode pengujian dan fungsionalitas selanjutnya [9]. Kode pengujian yang digunakan dapat berupa *integration testing* atau *scenario testing*. *Scenario testing* digunakan untuk memastikan bahwa fungsi pada aplikasi bekerja pada kondisi spesifik yang biasa digunakan dengan metode pengujian lain [21], sedangkan *integration testing* digunakan untuk memastikan bahwa sebuah fungsi pada aplikasi dapat bekerja sesuai ekspektasi secara keseluruhan. Metode ini meningkatkan kualitas produk perangkat lunak yang dikembangkan dan produktifitas pengembang.

Studi kasus pada tahun 2019 menyebutkan kelebihan saat menggunakan metode pengembangan perangkat lunak TDD sebagai berikut:

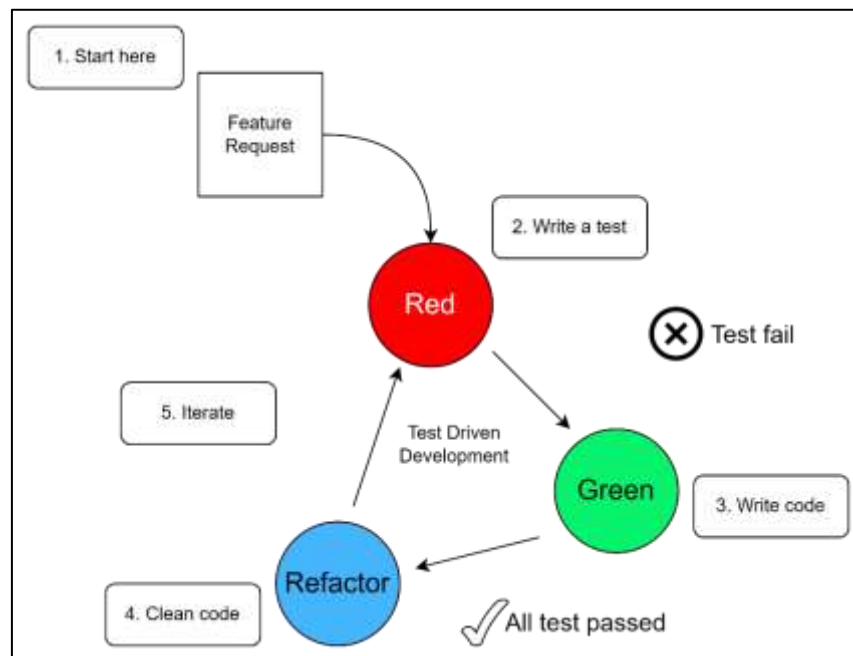
1. Membantu mencegah kecacatan produk
2. Membantu dokumentasi kode dengan contoh eksekusi
3. Membantu programmer untuk sangat memahami kode mereka
4. Mendukung refaktor sebagai kebutuhan dan perubahan desain
5. Mendorong desain yang lebih baik
6. Menyediakan peringatan awal terkait masalah desain

7. Membuat rangkaian uji regresi otomatis
8. Programmer mempelajari cara menulis jenis pengujian lain
9. Mendorong langkah kecil dan prinsip bahwa lebih baik untuk menjaga sistem tetap bekerja

Kemudian, pada studi yang sama juga menyebutkan kekurangan saat menggunakan metode pengembangan perangkat lunak TDD, yaitu sebagai berikut:

1. Sulit untuk dipelajari
2. Sulit untuk diaplikasikan kepada kode yang telah dibuat sebelumnya dari orang lain (*legacy code*)
3. Banyak kesalahpahaman yang mempersulit programmer untuk mempelajarinya.

*Test-Driven Development* (TDD) memiliki siklus pengembangan perangkat lunak seperti Gambar 2.1 berikut ini:



Gambar 2.1 SDLC *Test-Driven Development* (TDD) [22]

1. Langkah pertama yaitu membaca, memahami, dan memproses fitur atau bug yang diminta.

2. Berdasarkan kebutuhan yang diminta, unit pengujian akan dibuat dan dijalankan dengan kode pada langkah 3.
3. Membuat dan mengimplementasikan kode yang memenuhi kebutuhan pengujian. Saat dijalankan, semua pengujian harus lolos, jika tidak, langkah ini akan diulang hingga berhasil.
4. Apabila sudah berhasil, maka kode akan dibersihkan/dirapikan (*refactoring*).
5. Ulang dari langkah 1 dengan permintaan fitur atau bug yang baru.

Alur kerja *Test-Driven Development* (TDD) sebagaimana dilihat pada Gambar 2.1 disebut juga sebagai *Red-Green-Refactoring* (Merah-Hijau-Refaktor), yang merupakan status dari pengujian dalam setiap siklusnya [22].

Penelitian tugas akhir ini akan menggunakan siklus hidup pembangunan perangkat lunak (SDLC) *Test-Driven Development* (TDD) karena kebutuhan yang dapat bervariasi dari aplikasi yang akan dikembangkan baik dari fitur maupun kebutuhan utama. Dengan menggunakan TDD, aplikasi yang dikembangkan dapat diminimalisir adanya *error* atau *bug* dengan melakukan unit pengujian untuk setiap langkah pembangunannya[8]. Pengembangan aplikasi penilaian esai singkat otomatis ini juga tidak menggunakan *legacy code* pada sisi yang dikerjakan oleh penulis, sehingga tidak perlu khawatir dengan kekurangan pada penerapan TDD yang telah disebutkan sebelumnya tentang kesulitan penerapan TDD pada *legacy code* [7].

### 2.2.2 Use Case Diagram


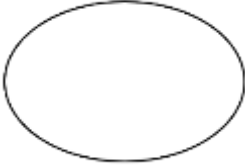
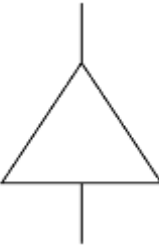



*Use Case Diagram* adalah salah satu diagram dalam *Unified Modeling Language*. *Use Case Diagram* merupakan diagram perilaku yang mendeskripsikan kebutuhan fungsional pada sebuah perangkat lunak. Diagram ini juga digunakan untuk memahami bagaimana sebuah sistem seharusnya bekerja [23]. Berikut ini merupakan manfaat dari digunakannya *Use Case Diagram*, yaitu:

1. Mendapatkan tampilan luar dari suatu sistem.
2. Mendapatkan seluruh kebutuhan sistem.
3. Mengenali faktor-faktor yang dapat mempengaruhi sistem secara eksternal maupun internal.

#### 4. Mendemonstrasikan interaksi antara sistem dan aktor.

Pada Tabel 2.2 dibawah ini, dapat diperhatikan nama dan deskripsi dari simbol yang akan digunakan dalam sebuah *Use Case Diagram*.

Tabel 2.2 Simbol *Use Case Diagram*

| Simbol  | Nama                | Deskripsi  |
|---|---------------------|--|
|    | Aktor               | Peranan eksternal yang berinteraksi dengan sistem.   |
|    | <i>Use Case</i>     | Fungsionalitas atau kebutuhan yang akan diimplementasikan pada sebuah sistem.  |
|   | <i>Generalisasi</i> | Merepresentasikan hubungan antara aktor dengan aktor, maupun <i>use case</i> dengan <i>use case</i> lainnya.   |
|  | Asosiasi            | Merepresentasikan komunikasi dua arah antara aktor dengan <i>use case</i> , setiap kasus dimulai dengan aktor utama yang harus mendapatkan respon dari <i>use case</i> .                                 |
|  | <i>Include</i>      | Merepresentasikan hubungan antara dua <i>use case</i> . Saat <i>use case</i> A membutuhkan <i>use case</i> B untuk menyelesaikan tugasnya, walau <i>use case</i> B dapat menyelesaikan tugasnya sendiri. |
|  | <i>Extend</i>       | Merepresentasikan hubungan antara dua <i>use case</i> . Saat <i>use case</i> A mungkin memerlukan <i>use case</i> B untuk menyelesaikan tugasnya, namun <i>use case</i> B tidak bisa ada sendiri.        |

Penelitian ini akan menggunakan *Use Case Diagram* untuk memperjelas skenario dan kebutuhan dari perilaku sistem yang akan dikembangkan. Hal ini dilakukan




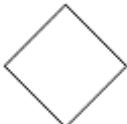




untuk memastikan bahwa sistem yang akan dikembangkan memenuhi skenario dan perilaku sistem yang telah diajukan.

### 2.2.3 Activity Diagram

Sama seperti *Use Case Diagram*, *Activity Diagram* merupakan salah satu diagram dalam *Unified Modeling Language*. *Activity Diagram* dapat menampilkan alur kegiatan dari sebuah sistem perangkat lunak [24]. *Activity Diagram* biasanya berisi aktifitas, transaksi, keputusan, swimlane, dan aktifitas paralel [25]. Berdasarkan itu, simbol, nama, dan deskripsi dari *Activity Diagram* dapat dilihat pada Tabel 2.3 berikut ini.

Tabel 2.3 Simbol *Activity Diagram*

| Simbol  | Nama              | Deskripsi  |
|---|-------------------|--|
|    | <i>Start</i>      | Menunjukkan kondisi awal / dimulainya sebuah aktifitas   |
|  | <i>End</i>        | Menunjukkan berakhirnya semua kondisi dari sebuah aktifitas  |
|  | Aktifitas         | Menunjukkan aktifitas dari sebuah proses   |
|  | Keputusan         | Menunjukkan kondisi percabangan dari 1 masukan dengan banyak keluaran  |
|  | <i>Swimlane</i>   | Mengorganisir aktifitas berdasarkan peran dari setiap proses   |
|  | Aktifitas Paralel | Menunjukkan proses yang berjalan secara paralel, biasanya terdiri dari dua aktifitas yang dijalankan bersamaan |

Penelitian ini akan menggunakan *Activity Diagram* untuk menggambarkan alur aktifitas dari sistem perangkat lunak yang akan dikembangkan.

#### 2.2.4 Website

*Website* adalah keseluruhan halaman-halaman web yang terdapat dari sebuah domain yang mengandung informasi [26]. Layanan yang terdapat pada *website* dapat diakses melalui jaringan internet, dengan permintaan layanan dieksekusi melalui sistem jarak jauh [27].

#### 2.2.5 JavaScript

*JavaScript* adalah bahasa pemrograman yang populer, digunakan bukan hanya untuk web sisi klien, tapi juga sisi *server* [28]. *JavaScript* juga memiliki tipe dinamis, yang artinya pembangun tidak perlu menspesifikasikan tipe dalam kodenya. Hal ini yang membuat IDE *JavaScript* sering gagal mensugestikan tipe yang akurat karena tipe dari elemen kode tidak diketahui hingga memiliki isi. Namun kekurangan ini bisa diatasi dengan membuat kode yang bagus dan tidak ambigu.

Pada penelitian tugas akhir ini, *JavaScript* akan digunakan untuk membangun aplikasi berbasis website penilaian esai singkat pada sisi klien dan juga sisi *server* dengan *Node.js*.

#### 2.2.6 Node.js

*Node.js* adalah *run timer* sisi server *JavaScript* yang digunakan pada *browser* untuk memproses HTTP. *Node.js* memungkinkan penulis untuk membuat *backend* dari sebuah *website* yang dapat menerima *request GET* dan *POST* menggunakan bahasa pemrograman *JavaScript* [29]. *Node.js* memiliki *package manager* yaitu NPM, gudang perangkat lunak terbesar di dunia dengan jutaan paket yang dapat digunakan pada perangkat lunak siapapun menjadi bantuan (*dependencies*).

Penelitian ini menggunakan *Node.js* sebagai *run timer* sisi *server* untuk aplikasi berbasis *website* yang akan dibangun karena dapat menangani lebih banyak permintaan dalam satuan waktu dibanding PHP [30]. Hal ini membuat *Node.js* lebih ideal untuk membangun *website* dengan masukan dan keluaran yang intens. Selain itu, *Node.js* juga *developer friendly*, yang mana mudah dipahami karena cara penggunaannya yang konsisten.

### 2.2.7 Express.js

Pada teori sebelumnya, dapat diketahui bahwa *Node.js* merupakan *run timer JavaScript* dan dapat menerima permintaan HTTP *GET* dan *POST*. *Express.js* merupakan paket NPM berupa *framework* untuk mempermudah penggunaan modul HTTP pada *Node.js* [31] sehingga penulisan kode untuk melakukan permintaan HTTP dapat dilakukan lebih cepat. Sebagai *framework* untuk menangani permintaan HTTP, performa *Express.js* terbilang cepat yaitu hanya memerlukan waktu 3 ms untuk memproses permintaan *Object Relational Model* [17], dan memerlukan waktu rata-rata 17 ms untuk memproses halaman dinamis dari permintaan sebanyak 200 pengguna.

Atas alasan tersebut, perangkat lunak yang akan dikembangkan dalam penelitian ini akan menggunakan *Express.js* untuk menangani permintaan HTTP sebagai *framework* dari *Node.js* karena kemampuan paket NPM ini diperlukan untuk memproses nilai dan jawaban pada aplikasi *website* penilaian esai singkat.

### 2.2.8 JavaScript Object Notation

*JavaScript Object Notation* (JSON) adalah format data dengan penulisan seperti tipe data objek pada bahasa pemrograman *JavaScript*. JSON merupakan format data berisi objek atau kumpulan objek-objek dalam suatu objek yang ringan, mudah dibaca dan ditulis oleh manusia, serta dibuat dan diterjemahkan dengan komputer. Format data ini merupakan bagian dari bahasa pemrograman *JavaScript* [32]. JSON umumnya berisikan sebuah objek yang dapat berisi banyak objek lain. Dalam format dokumen ini, objek harus dimulai dan diakhiri oleh kurung keriting (`{}`), kemudian setiap objek dalam JSON dipisah dengan simbol koma (,) setelah kurung keriting tutup (`}`) [32]. Untuk menggunakan dokumen JSON pada aplikasi *website* khususnya yang menggunakan *JavaScript*, objek dapat diakses dengan menggunakan dokumen JSON dan menggunakan simbol titik (.) untuk mengakses setiap sub-objeknya.

Pada aplikasi web yang akan dikembangkan pada penelitian ini, akan ada fitur untuk mengunggah seluruh jawaban pelajar berupa berkas CSV ke aplikasi *website*

penilaian esai. Berkas CSV kemudian akan diubah menjadi bentuk JSON untuk selanjutnya diproses kedalam penilaian.

### 2.2.9 Esai Singkat

Esai merupakan karangan yang membahas suatu masalah berdasarkan sudut pandang penulis, pada kasus ini yaitu jawaban dari pertanyaan yang diberikan oleh pengajar menurut pelajar. Esai merupakan salah satu metode penilaian untuk mengukur tingkat pemahaman pelajar pada suatu topik yang kerap diberikan pada saat ujian [33]. Esai singkat menurut Steven Burrows pada tahun 2014 merupakan sebuah teks deskriptif tentang suatu topik khusus yang terdiri dari sebuah frasa atau paling banyak 100 kata [34].

### 2.2.10 Python

*Python*, seperti *JavaScript*, merupakan salah satu bahasa pemrograman yang banyak digunakan oleh pembangun perangkat lunak di bidang teknologi. Bedanya, *Python* lebih banyak digunakan untuk sains data dengan kapabilitasnya untuk pembelajaran mesin [35]. *Python* merupakan bahasa pemrograman yang banyak dipilih bagi praktisi pemrosesan bahasa alami, yang merupakan cabang dari kecerdasan buatan. *Python* memiliki *library* yang dapat mendukung pemrosesan bahasa alami yang mencakup banyak bahasa manusia seperti *Stanza* [36].

Aplikasi penilaian esai singkat yang akan dikembangkan penulis akan menggunakan model terlatih berbasis *Python* untuk melakukan proses penilaian dan perbandingan jawaban.

### 2.2.11 System Usability Scale

Dalam membangun sebuah sistem perangkat lunak, diperlukan adanya skala standar yang dapat digunakan untuk memastikan bahwa sistem yang dikembangkan dapat digunakan oleh pengguna sebaik-baiknya. *System Usability Scale* (SUS) adalah kuesioner yang digunakan untuk mengukur tingkat kemudahan perangkat lunak saat digunakan [11]. SUS adalah alat psikometrik gratis yang digunakan diseluruh dunia dengan keabsahan dan keandalan tinggi [12].

Pada penelitian ini, penulis menggunakan SUS untuk mengukur tingkat kemudahan pengguna dalam menggunakan aplikasi penilaian esai singkat otomatis berbasis web melalui antarmuka yang akan disediakan untuk mengoperasikan aplikasi yang akan dibangun.

Kuesioner SUS akan diajukan kepada sejumlah pengguna relawan yang tertarik untuk menggunakan aplikasi penilaian esai otomatis berbasis web yang akan dikembangkan pada penelitian tugas akhir ini. Kemudian hasil dari kuesioner SUS ini akan berupa rata-rata skor dari semua responden dengan rentang nilai dari 0 s/d 100. Nilai prinsip SUS adalah untuk menyediakan skor referensi dari pandangan peserta terhadap kegunaan sebuah produk / sistem [37].

Nilai akan diperoleh dengan cara memisahkan skor pertanyaan dengan nomor ganjil dari nomor genap dengan kalkulasi pada rumus (1) berikut ini :

$$\bar{x} = \frac{\sum(2.5 \times (\frac{a-1}{5-b}))}{n}$$

Rumus perhitungan SUS (1)

Keterangan:

- $\alpha$  : skor pertanyaan bernomor ganjil  
 $b$  : skor pertanyaan bernomor genap  
 $\bar{x}$  : skor rata-rata  
 $n$  : jumlah responden

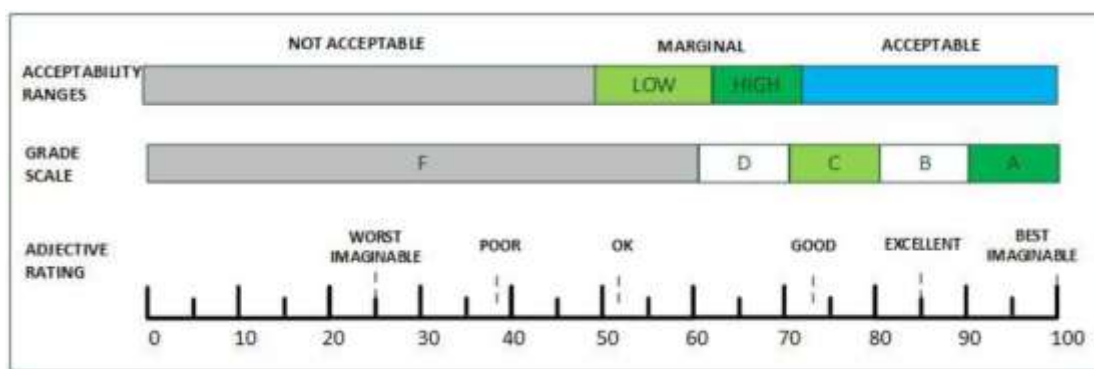
Kemudian pada Tabel 2.4 berikut ini, dapat dilihat interpretasi dari setiap nilai yang diperoleh kalkulasi pengujian SUS.

Tabel 2.4 Standar Hasil SUS [14]

| Skor SUS  | Nilai | Sifat Nilai |
|-----------|-------|-------------|
| >80.3     | A     | Sangat Baik |
| 74 – 80.3 | B     | Baik        |

| Skor SUS | Nilai | Sifat Nilai  |
|----------|-------|--------------|
| 68 - 74  | C     | Oke          |
| 51 – 68  | D     | Buruk        |
| <51      | F     | Sangat Buruk |

Melalui standar hasil SUS pada Tabel 2.4, aplikasi penilaian esai otomatis berbasis web yang dikembangkan sebaiknya meraih minimal nilai C untuk dapat dinyatakan berhasil dan dapat digunakan oleh banyak pengguna sebagaimana terdapat pada Gambar 2.2 berikut ini [14].



Gambar 2.2 SUS Score Percentile Rank [14]

### 2.2.12 Black Box Testing

Teknik pengujian selanjutnya yaitu teknik pengujian *Black Box*. Teknik pengujian aplikasi ini membahas aplikasi perangkat lunak dari sisi luar seperti tampilan, fungsionalitas, masukan, dan keluaran [15].

Aplikasi yang akan dikembangkan dari penelitian ini akan diuji menggunakan pengujian *Black Box*. Aplikasi akan diuji oleh peneliti untuk menguji masukan dan keluaran dari aplikasi ini dengan maksud untuk memastikan bahwa aplikasi yang dibangun bekerja semestinya.

### 2.2.13 Integration Testing

Untuk meningkatkan kepastian bahwa suatu program berfungsi sebagaimana diinginkan, perlu proses validasi untuk melakukannya. Proses validasi dapat dilakukan menggunakan *Integration Testing* karena dapat membuktikan bahwa

sebuah fungsi pada program telah memenuhi spesifikasinya [10]. *Integration Testing* dilakukan dengan memastikan bahwa komponen-komponen pada program saling cocok dan memberikan hasil sesuai harapan [38].

#### **2.2.14 Peramban Web**

Dengan semakin bertambahnya kemampuan teknis internet, bertambah juga kebutuhan untuk sebuah aplikasi web yang mempengaruhi berubahnya cara sebuah halaman web untuk menampilkan sebuah data dan informasi [39].

*JavaScript* telah mencapai platform web secara luas dan menjadi bahasa pemrograman multifungsi dengan adanya ES6 yang diimplementasikan pada mesin peramban web sejak tahun 2017 [40].

#### **2.2.15 Peluncuran Perangkat Lunak**

Peluncuran (*deployment*) perangkat lunak merupakan proses eksekusi dari sebuah perangkat lunak. Peluncuran adalah aktifitas dari sebuah perangkat lunak setelah proses pembangunan perangkat lunak [41]. Peluncuran dilakukan untuk mendapatkan timbal balik berupa perbaruan, konfigurasi ulang, adaptasi, penonaktifan, dan peluncuran ulang pada sebuah perangkat lunak [42].