BAB 2
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Tinjauan Pustaka dalam penelitian ini merupakan referensi penulisan dalam melakukan penelitian Pengujian Black Box Menggunakan Metode Decision Table Testing pada Google Speech-to-Text. Referensi ditunjukkan pada tabel 2.1.

Tabel 2.1 Tinjauan Pustaka

No	Nama Pengarang	Tahun	Judul	Objek	Bahasa pemrograman/ Teknologi/Metode
1	Taufik Hidayat, Mahmudin Muttaqin	2018	Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis	Aplikasi Sistem Informasi Pendaftaran dan Wisuda Online	Black box testing metode Equivalence Partitioning dan Boundary Value Analysis
2	M. Sidi Mustaqbal, Roeri Fajri Firdaus, Hendra Rahmadi	2016	Pengujian Aplikasi Menggunakan Black Box Testing Boundary Value Analysis (Studi Kasus : Aplikasi Prediksi Kelulusan SNMPTN)	Aplikasi Prediksi Kelulusan SNMPTN	Black box testing metode boundary value analysis
3	Tri Snadhika Jaya	2018	Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus : Kantor Digital Politeknik Negeri Lampung)	Aplikasi Kantor Digital Politeknik Negeri Lampung	Black box testing metode boundary value analysis
4	Youllia Indrawaty, Andriana Zulkarnain, Reza Rianto	2018	Pengembangan Pembelajaran Pengenalan Kata Dalam Bahasa Indonesia Menggunakan Multimedia Interaktif dan Speech Recognition	Aplikasi Sistem Informasi Sekolah	Black box testing metode equivalence partitions

Lanjutan Tabel 2.1 Tinjauan Pustaka

No	Nama Pengarang	Tahun	Judul	Objek	Bahasa pemrograman/ Teknologi/Metode
5	Nadya Safitru, Rully Pramudita	2018	Pengujian Black Box Menggunakan Metode Cause Effect Relationship Testing	Aplikasi Revo Uninstaller	Black box testing metode cause effect

Taufik Hidayat dan Mahmudin Muttaqin (2018) melakukan penelitian terhadap sistem informasi pendaftaran dan pembayaran wisuda online dan mengambil kasus pada bagian login peserta wisuda, *input* biodata, dan edit judul skripsi yang direpresentasikan dalam bentuk tabel.

M. Sidi Mustaqbal, Roeri Fajri Firdaus, dan Hendra Rahmadi (2016) melakukan *black box testing* terhadap aplikasi prediksi kelulusan SNMPTN menggunakan metode Boundary Value Analysis. Hasil dari penelitian ini, berdasarkan pengujian terhadap fungsi tambah kelas yang terdiri dari 2 kolom *input* keyboard yaitu Nama Kelas, dan Tahun Ajaran, diperoleh hasil masih perlu adanya penyempurnaan dengan menambahkan fungsi validasi data.

Tri Snadhika Jaya (2018) melakukan *black box testing* terhadap Kantor Digital Politeknik Negeri Lampung menggunakan metode Boundary Value Analysis. Hasil dari penelitian ini, berdasarkan pengujian terhadap fungsi surat masuk yang terdiri dari 9 *field entry*, diperoleh hasil tingkat kesuksesan dari kesembilan field sejumlah 92,67% dengan 6 field diantaranya menunjukkan tingkat kesuksesan sebesar 100%, 3 sisanya sebesar 75%.

Youllia Indrawaty, Andriana Zulkarnain, dan Reza Rianto (2018) membuat aplikasi yang menggunakan teknologi *speech recognition* dengan menggunakan algoritma *levenshtein distance*. Hasil dari penelitian ini diperoleh hasil 49% responden anak-anak *preschool* Pendidikan Anak Usia Dini (PAUD) memilih setuju aplikasi pengembangan pembelajaran pengenalan kata dalam bahasa Indonesia menggunakan multimedia interaktif dan *speech recognition* menarik, 32,5% memilih biasa saja, dan 18,5% memilih sangat setuju.

Nadya Safitru dan Rully Pramudita (2018) melakukan *black box testing* dengan menggunakan metode Cause Effect Relationship terhadap aplikasi Revo Uninstaller. Hasil dari penelitian ini menunjukkan bahwa aplikasi revo *uninstaller* yang menjadi contoh aplikasi yang diujikan memiliki fungsi-fungsi yang sudah berjalan dengan baik sesuai dengan yang diharapkan.

2.2 Dasar teori

2.2.1 *Software Testing*

Pengujian perangkat lunak (*software testing*) adalah aktivitas-aktivitas yang bertujuan untuk mengevaluasi atribut-atribut atau kemampuan sebuah program atau sistem dan penentuan apakah sesuai dengan hasil yang diharapkan atau tidak. Pengujian perangkat lunak spesifiknya adalah proses mengeksekusi suatu program untuk menemukan *bug* dari perangkat lunak. Pengujian yang sukses adalah bila pengujian yang dilakukan berhasil menemukan suatu kesalahan yang awalnya tidak ditemukan. *Software testing* terdiri dari 8 jenis, yaitu :

1. *Performance Testing*

Performance Testing adalah *integrationi* dan *usability test* yang menentukan apakah *system* atau *subsystem* dapat memenuhi kriteria kinerja berbasis waktu seperti *response time* atau *throughput*

2. *System Testing*

   System Testing adalah *integration test* dari *behavior* seluruh sistem atau *independent subsystem*. *System testing* biasanya dilakukan pertama kali oleh pengembang atau personil pengujian untuk memastikan bahwa keseluruhan sistem tidak berfungsi dan bahwa sistem telah memenuhi persyaratan pengguna (*user requirement*)

3. *Unit Testing*

   Unit Testing adalah proses metode pengujian *individual, class,* atau komponen sebelum mereka terintegrasi dengan perangkat lunak lainnya. Tujuannya adalah untuk mengidentifikasi dan memperbaiki kesalahan sebanyak mungkin sebelum modul-modul digabungkan menjadi unit perangkat lunak yang lebih besar.

4. *Integration Testing*

   Integration Test proses evaluasi *behavior* dari kelompok *method* atau *class.* Tujuannya adalah untuk mengidentifikasi kesalahan yang tidak dapat dideteksi oleh unit testing seperti *Interface Incompability, Parameter Values, Run-Time Exceptions,* dan *Unexpected State Interactions.*

5. *Usability Testing*

Usability Test adalah test untuk menentukan apakah *method, clas, subsystem,* atau sistem telah memenuhi persyaratan pengguna. Pengujian ini dilakukan untuk mendapatkan *feedback* yang cepat dalam meningkatkan *interface* dan mengkoreksi kesalahan dalam komponen perangkat lunak.

6. *Smoke Testing*

   Smoke Testing adalah *system test* yang dilakukan setiap hari atau beberapa kali per minggu. Pengujian ini dilakukan setelah sebuah perangkat lunak telah selesai di *build* untuk memastikan bahwa fungsi penting dari program telah bekerja dengan baik.

7. *Stress Testing*

   Stress Testing adalah pengujian yang biasanya dilakukan dalam membuat sebuah *website* untuk mengetahui seberapa kuat server *website* dalam menampung pengunjung.

8. *User Acceptence Test (UAT)*

   User Acceptance Test digunakan untuk menentukan apakah sistem yang dikembangkan telah memenuhi kebutuhan pengguna atau belum. Pengujian ini dilakukan setelah rangkaian pengujian seperti *Unit Testing, Integration Testing,* dan *System Testing* selesai dan menggunakan metode *Black Box Testing*.

2.2.2 *Black Box Testing*

Black Box testing berfokus pada pengujian dari masing-masing spesifikasi fungsional perangkat lunak. Seorang *tester* dapat

mendefinisikan kumpulan kondisi *input* dan melakukan pengetesan pada fungsionalitas perangkat lunak (Mustaqbal, 2015). Metode *Black Box testing* terdiri atas beberapa metode, antara lain *Equivalence Partitioning, Boundary Value Analysis, State Transition Testing,* dan *Decision Table Testing*.

Black Box Testing merupakan metode pengujian perangkat lunak yang digunakan untuk menguji sebuah perangkat lunak tanpa mengetahui struktur internal kode atau program. Dalam pengujiannya, penguji menyadari apa yang harus dilakukan oleh program, tapi tidak memiliki pengetahuan tentang bagaimana melakukannya. Kelebihan black box testing yaitu :

1. Efisien untuk segmen kode besar.
2. Akses kode tidak diperlukan
3. Pemisahan antara perspektif pengguna dan pengembang

Selain memiliki kelebihan, *black box testing* juga memiliki kelemahan, yaitu :

1. Cakupan terbatas karena hanya sebagian kecil dari skenario pengujian yang dilakukan.
2. Pengujian tidak efisien karena keberuntungan *tester* dari pengetahuan tentang perangkat lunak internal

2.2.3 *Decision Table Testing*

Decision Table Testing adalah salah satu metode *Black Box Testing* yang digunakan untuk menguji perilaku sistem dengan

beberapa kombinasi *input* yang berbeda. Metode ini biasa disebut sebagai tabel *Cause-Effect*. *Decision Table Testing* dilakukan ketika sistem memiliki berbagai variasi nilai input yang berbeda yang tidak bisa diuji melakukan metode *boundary value analysis*, ataupun *equivalent partitioning.*

Contoh penerapan *Decision Table Testing* bisa ditemukan pada fungsional program untuk melakukan upload foto dengan ketentuan :

1. Hanya bisa mengunggah file format ".jpg".
2. file kurang dari 32 kb
3. resolusi gambar 137*177

Maka hasil *Decision Table Testing*-nya adalah seperti pada gambar 2.1 di bawah ini :

Condition	**Case 1**	**Case 2**	**Case 3**	**Case 4**	**Case 5**	**Case 6**	**Case 7**	**Case 8**
Format	.jpg	.jpg	.jpg	.jpg	Not .jpg	Not .jpg	Not .jpg	Not .jpg
Size	Less than 32kb	Less than 32kb	>= 32kb	>= 32kb	Less than 32kb	Less than 32kb	>= 32kb	>= 32kb
resolution	137*177	Not 137*177	137*177	Not 137*177	137*177	Not 137*177	137*177	Not 137*177
Output	Photo uploaded	Error message resolution mismatch	Error message size mismatch	Error message size and resolution mismatch	Error message for format mismatch	Error message format and resolution mismatch	Error message for format and size mismatch	Error message for format, size, and resolution mismatch

***Gambar 2.1 Decision Table Testing Upload* Gambar**

2.2.4 *Google Speech-to-Text API*

Google Speech-to-Text memungkinkan *developer* untuk mengonversi *audio* menjadi teks dengan menerapkan model jaringan neural yang andal ke dalam API yang mudah digunakan. API ini mengenali 120 bahasa dan varian untuk mendukung basis pengguna global dengan cara mengaktifkan melalui perintah dan kontrol suara. API ini memproses *streaming real time* atau *audio* rekaman menggunakan *machine learning* Google.

2.2.5 Zona dalam Komunikasi

Menurut buku yang ditulis oleh Nageshwar Rao (2009) yang diterbitkan oleh Himalaya Publishing House di Mumbai, India, terdapat empat zona jarak dalam komunikasi, yaitu :

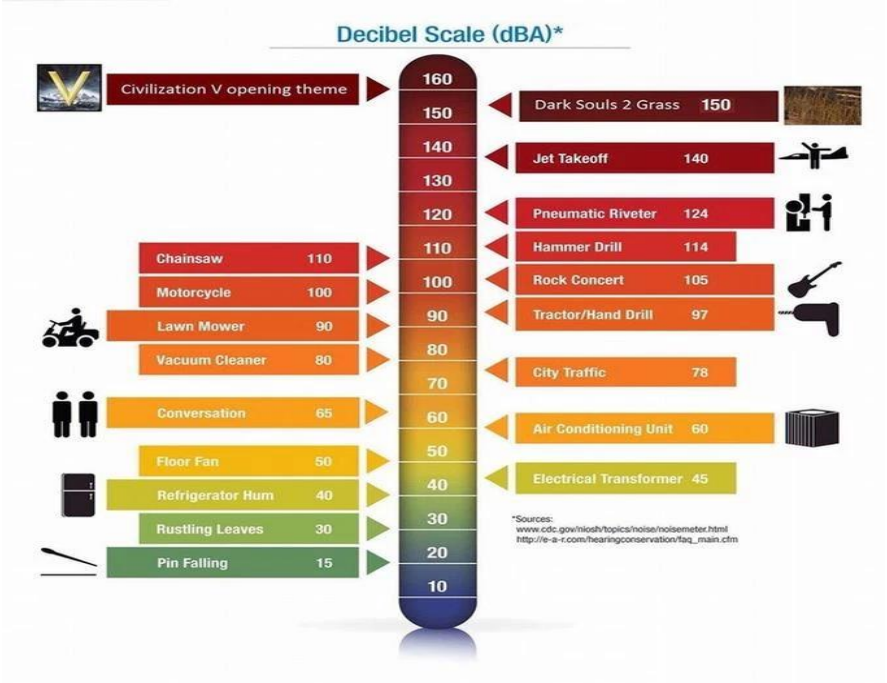
1. Zona Intim yaitu jarak 0-1,5 kaki (45.5 cm).
2. Zona Personal yaitu jarak 1,5-4 kaki (124 cm).
3. Zona Sosial yaitu jarak 4-12 kaki (378 cm).
4. Zona Publik yaitu jarak 12 kaki dan seterusnya.

2.2.6 *Sound Level Meter*

Sound Level meter adalah suatu alat yang digunakan untuk pengukuran suatu intensitas suara dengan menggunakan satuan desibel (dB). Penggunaan Sound Level Meter dilakukan dengan cara diletakkan setidaknya 5 cm dari sumber suara. Sound Level Meter sangat diperlukan terutama untuk lingkungan industri yang memiliki tingkat noise cenderung tinggi untuk mengetahui pengaruh terhadap

lingkungan sekitar. Sound Level Meter berfungsi untuk mengukur kebisingan (*noise*) antara 30-130 dB dalam satuan dBA dari frekuensi antara 20-20.000Hz.

2.2.7 *Noise*



Gambar 2.2 Sound Pressure Level

Sound Pressure Level menunjukkan seberapa besar perubahan tekanan yang dialami oleh suatu medium atau perantara (pada umumnya udara) dari kondisi setimbangnya. Misalnya jika kita memberikan perubahan sebesar 20 mikro Pascal, maka telinga akan mempersepsikannya sebagai suara dengan level 0 dB, sedangkan bila perubahannya sebesar 100.000.000 mikro Pascal, akan dipersepsi sebagai suara dengan level 140 dB

Gambar 2.1 merupakan *Sound Pressure Level* dalam skala satuan desibel(dB) yang disertai dengan contoh suaranya pada titik tertentu yang

dijadikan acuan. Pada gambar 2.1 diketahui bahwa suara percakapan manusia ada di sekitar 65 dB yang mana merupakan kategori sedang bagi pendengaran manusia. Pada skala mendekati 80 keatas yaitu kondisi lalu lintas dalam kota merupakan kategori sangat keras.