BAB 2
LANDASAN TEORI

2.1 Managemen Waktu

Manajemen waktu adalah proses yang melibatkan perencanaan dan pengendalian penggunaan waktu dalam berbagai aktivitas untuk meningkatkan produktivitas sekaligus mencapai tujuan dengan lebih efisien [2]. Secara umum, manajemen waktu mencakup pengelolaan diri yang dilakukan secara logis dan dewasa secara psikologis, tanpa melibatkan emosi, sehingga waktu yang dimiliki dapat dimanfaatkan secara optimal. Dengan manajemen waktu yang baik, seseorang dapat menyelesaikan tugas dengan tepat waktu serta menghasilkan output yang memuaskan. Selain itu, penerapan manajemen waktu membantu individu menjadi lebih produktif dan memiliki pola rutinitas yang lebih terstruktur [8].

Terdapat empat komponen utama yang memengaruhi keberhasilan manajemen waktu [9]:

1. Penetapan tujuan dan prioritas, yakni mengidentifikasi tujuan yang ingin dicapai serta menentukan prioritas tugas yang mendukung tercapainya keberhasilan.

2. Perencanaan dan penjadwalan yang melibatkan kebiasaan seperti menyusun daftar pekerjaan, merancang aktivitas, dan menjadwalkan kegiatan sesuai kebutuhan.

3. Preferensi organisasi, yang merujuk pada kebiasaan seseorang untuk menjaga kerapihan dan keteraturan baik dalam lingkungan kerja maupun dalam pendekatan penyelesaian tugas.

4. Persepsi kendali atas waktu, yaitu keyakinan seseorang terhadap kemampuannya untuk mengelola waktu dengan baik, termasuk kepercayaan diri dalam memanfaatkan waktu secara efisien dan menangani berbagai aktivitas secara efektif.

2.2 Priority Scheduling Algorithm

Algoritma *Priority Scheduling* merupakan algoritma penjadwalan yang mengurutkan proses berdasarkan prioritas, di mana setiap proses dijadwalkan

dengan nomor prioritas tertentu. Dalam algoritma ini, proses dengan prioritas tertinggi akan diproses lebih dulu. Jika terdapat proses dengan prioritas yang sama, maka proses yang pertama kali datang akan diproses terlebih dahulu [10]. Setiap proses diberikan prioritas, dan penjadwalan akan memilih proses dengan prioritas tertinggi untuk dieksekusi. Menurut Allwin M. Simarmata dan Mawaddah Harahap, prioritas dalam proses dapat ditentukan berdasarkan beberapa kriteria [7]:

1. *Time limit*
2. *Memory requirement*
3. Akses *file*
4. Perbandingan antara I/O *Burst* dengan CPU *Burst*
5. Tingkat kepentingan proses

Teknik implementasi algoritma *Priority Scheduling* dapat diklasifikasikan ke dalam tiga jenis, yaitu [11]:

1. *Preemptive Priority Scheduling*

   Pada jenis ini, tugas dengan prioritas lebih tinggi dapat mengambil alih eksekusi tugas yang sedang berjalan. Proses implementasinya melibatkan penggunaan *queue* prioritas serta mekanisme *interrupt* untuk memungkinkan pergantian tugas.

2. *Non-preemptive Priority Scheduling*

   Pada pendekatan ini, tugas yang sedang dieksekusi tidak dapat digantikan oleh tugas lain, meskipun memiliki prioritas lebih tinggi. Pelaksanaannya menggunakan *queue* prioritas yang hanya mengevaluasi tingkat prioritas setelah tugas yang sedang berjalan selesai.

3. *Dynamic Priority Scheduling*

   Dalam metode ini, prioritas sebuah tugas dapat berubah selama proses eksekusi. Teknik ini diimplementasikan dengan algoritma yang memungkinkan penyesuaian prioritas, seperti mekanisme *aging*.

Contoh analisis algoritma *Priority Scheduling* dapat dilihat pada Tabel 2.1.

Tabel 2.1. Contoh analisis algoritma *priority scheduling*

Process	*Duration*	*Priority*	*Arrival Time*
P1	6	4	0
P2	8	1	0
P3	7	3	0
P4	3	2	0

Sumber: Simarmata dan Harahap (2019) [7]

Dari Tabel 2.1, dapat dijelaskan bahwa algoritma *priority scheduling* mengeksekusi proses berdasarkan prioritas tertinggi. Proses pertama yang dieksekusi adalah P2, karena memiliki prioritas tertinggi, dengan waktu penyelesaian sebesar 8 satuan waktu. Setelah P2 selesai, eksekusi dilanjutkan dengan P4, yang memiliki prioritas tertinggi berikutnya, dengan durasi pengerjaan sebesar 3 satuan waktu. Selanjutnya, proses P3 dieksekusi dengan waktu pengerjaan selama 7 satuan waktu. Terakhir, proses P1 dikerjakan dengan durasi sebesar 6 satuan waktu.

Pada algoritma *Priority Scheduling*, eksekusi setiap tugas dilakukan berdasarkan tingkat nilai prioritas yang dimiliki. Proses berikutnya tidak akan dijalankan sebelum proses sebelumnya selesai dieksekusi [11]. Evaluasi algoritma *Priority Scheduling* dilakukan menggunakan dua parameter utama, yaitu *Turn Around Time* (TAT) dan *Waiting Time* (WT). Sebelum evaluasi dilakukan, proses penentuan prioritas setiap tugas harus diselesaikan terlebih dahulu dengan memberikan bobot nilai untuk setiap kriteria yang relevan. Penentuan bobot ini dilakukan berdasarkan metode yang dijelaskan oleh Kustanto et al. (2024) [12]. Setelah bobot prioritas berhasil ditentukan, evaluasi *Priority Scheduling* dapat dilakukan melalui langkah-langkah berikut:

1. Menentukan *Completion Time* (CT)

   Completion Time (CT) adalah waktu di mana suatu proses selesai dieksekusi sepenuhnya. CT dihitung dengan menjumlahkan waktu kedatangan (*Arrival Time* atau AT) dengan durasi proses (*Burst Time* atau BT), serta waktu tunggu yang dialami proses sebelumnya (jika ada).

2. Menghitung *Turn Around Time* (TAT)

   TAT adalah selisih antara *Completion Time* (CT) dan *Arrival Time* (AT) untuk setiap proses. Perhitungan TAT dinyatakan pada Rumus 2.1.

$$TAT = CT - AT \tag{2.1}$$

Nilai TAT menggambarkan total waktu yang dibutuhkan sejak proses datang hingga proses selesai dieksekusi.

3. Menghitung *Waiting Time* (WT)

   WT adalah selisih antara *Turn Around Time* (TAT) dan *Burst Time* (BT). Perhitungan TAT dinyatakan pada Rumus 2.2.

$$WT = TAT - BT \tag{2.2}$$

   WT menunjukkan total waktu tunggu yang dialami suatu proses sebelum dapat dieksekusi sepenuhnya.

4. Menghitung Rata-rata TAT dan WT

   Setelah nilai TAT dan WT dihitung untuk semua proses, rata-rata dari masing-masing parameter dapat dihitung menggunakan Rumus 2.3 dan Rumus 2.4.

$$Rata - rata_{TAT} = \frac{\sum TAT}{n} \tag{2.3}$$

$$Rata - rata_{WT} = \frac{\sum WT}{n} \tag{2.4}$$

   dengan n adalah jumlah total proses yang dieksekusi.

Proses evaluasi ini dapat dilakukan dengan merujuk pada penelitian "*Comparison of CPU Scheduling Algorithms: FCFS, SJF, SRTF, Round Robin, Priority Based, and Multilevel Queuing*" oleh Sajeewa Pemasinghe dan Samantha Rajapaksha [13]. Penelitian tersebut menjelaskan berbagai perbandingan algoritma penjadwalan CPU, termasuk *Priority Scheduling*, serta metrik evaluasi seperti TAT dan WT yang digunakan untuk menilai efisiensi algoritma.

Salah satu kelemahan utama dalam algoritma *Priority Scheduling* adalah potensi terjadinya *indefinite blocking* atau *starvation*. Kondisi ini terjadi ketika suatu proses dengan prioritas rendah tidak mendapatkan kesempatan untuk dieksekusi karena selalu terdapat proses lain dengan prioritas yang lebih tinggi. Situasi ini dapat menyebabkan proses prioritas rendah terus-menerus tertunda hingga tidak pernah dieksekusi sama sekali [14]. Untuk mengatasi permasalahan

tersebut, teknik *aging* dapat diterapkan. Teknik ini bekerja dengan cara meningkatkan prioritas setiap proses secara bertahap seiring waktu, sehingga proses dengan prioritas rendah tetap memiliki peluang untuk dieksekusi.

Menurut, Yanwari et al., (2023) *priority* juga terdapat sebuah *constraint* (batasan) untuk dapat menentukan *task* prioritas lebih spesifik yang terdapat dua tipe *constraint* yaitu *Soft Constraint* dan *Hard Constraint* [11]. *Hard constrains* berperan sebagai kondisi yang wajib dipenuhi dan *Soft Constraint* berperan sebagai kondisi yang diinginkan tetapi tidak wajib dipenuhi. Kedua batasan ini memiliki perlakuan yang berbeda dalam proses *scheduling*. Beban dari *Soft Constraint* hanya akan mempengaruhi beban *queue* dalam poin beban positif saja, sedangkan beban dari *Hard Constraint* mempengaruhi beban *queue* pada poin beban positif dan negatif tergantung pada apakah item dalam *queue* dapat memenuhi *Hard Constraint* atau tidak [11].

2.3 PHP

PHP adalah sebuah bahasa pemrograman berbasis *server-side scripting*, yang berfungsi untuk merancang elemen-elemen yang dapat digunakan dalam pengembangan sebuah situs *web*. Konsep *server-side scripting* mengacu pada fungsi skrip yang terdiri dari serangkaian instruksi yang sepenuhnya dijalankan di server, tetapi terintegrasi dalam dokumen HTML biasa [15].

Sebagai bahasa pemrograman yang populer untuk pengembangan aplikasi berbasis *web*, PHP sering digunakan untuk membangun sistem informasi berbasis *web*. PHP memungkinkan komunikasi dengan *database*, menghasilkan konten yang bersifat dinamis, serta mendukung integrasi dengan berbagai teknologi *web* lainnya [16].

2.4 CodeIgniter

CodeIgniter adalah *framework* PHP bersifat *open-source* yang dirancang untuk pengembangan *website* secara ringan dan sederhana [17]. Sebagai *framework* berbasis PHP, CodeIgniter bertujuan untuk mempermudah pengembangan aplikasi *web*. *Framework* ini dipilih karena ukurannya yang kecil, dokumentasi yang jelas, serta dukungan komunitas yang luas [18].

Framework ini menggunakan pola MVC (*Model*, *View*, *Controller*) untuk menciptakan *website* yang lebih dinamis. CodeIgniter memungkinkan

pengembang untuk membangun aplikasi *web* dengan cepat dan efisien sejak awal. Selain menciptakan *website* yang lebih dinamis, *framework* ini juga mendukung pengembangan aplikasi *web* yang ringan dan responsif. Dengan dokumentasi lengkap yang mencakup berbagai contoh implementasi kode, CodeIgniter memudahkan proses pengembangan web secara keseluruhan [19].

2.5 Black Box Testing

Black box testing adalah metode pengujian perangkat lunak yang berfokus pada fungsi dan aspek eksternal aplikasi, seperti tampilan *website*, fitur-fitur yang tersedia dalam *website*, serta kesesuaian alur fungsi aplikasi tanpa memeriksa detail internal dari kode perangkat lunaknya [20].

Metode ini digunakan untuk mengevaluasi hasil eksekusi sistem berdasarkan masukan tertentu (*test case*) guna memastikan bahwa fungsionalitas sistem sudah sesuai dengan spesifikasi yang ditetapkan (*requirement*). *Black box* testing lebih berorientasi pada antarmuka atau tampilan aplikasi serta menguji fungsionalitas yang digunakan oleh pengguna untuk memastikan jalannya aplikasi sesuai kebutuhan [21].

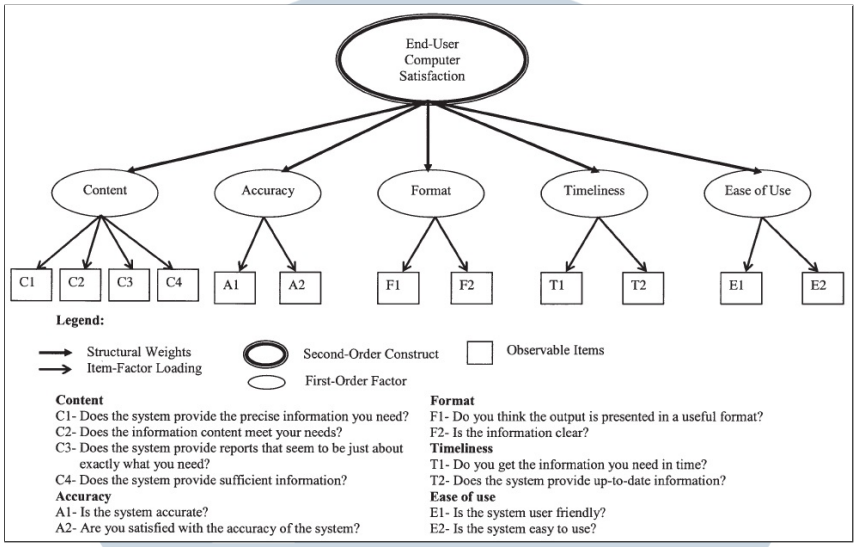
Selain itu, *black box* testing bertujuan untuk mengidentifikasi kesalahan dalam sistem aplikasi, seperti kesalahan fungsi, fitur yang hilang, atau ketidaksesuaian lainnya. Beberapa kesalahan yang dapat ditemukan melalui metode ini meliputi [22]:

1. Kesalahan atau hilangnya fungsi tertentu.
2. Masalah pada *interface* aplikasi.
3. Gangguan pada struktur data atau akses ke *database*.
4. Penurunan performa aplikasi.
5. Kesalahan dalam inisialisasi maupun proses akhir.

2.6 End User Computing Satisfaction (EUCS)

End User Computing Satisfaction (EUCS) adalah metode yang dikembangkan oleh Doll dan Torkzadeh untuk mengukur sejauh mana pengguna merasa puas terhadap sistem informasi dengan mengevaluasi lima dimensi utama: *content* (konten), *accuracy* (akurasi), format, *timeliness* (ketepatan waktu), dan

ease of use (kemudahan penggunaan) [23]. Ilustrasi dari model pengukuran EUCS dapat dilihat pada Gambar 2.1.



Gambar 2.1. Model pengukuran *end user computing satisfaction*

Sumber: Doll et al., (2004) [24]

Evaluasi sistem informasi menggunakan EUCS merupakan metode evaluasi bersifat subjektif, dimana setiap faktor atau dimensi pada EUCS mencerminkan keberhasilan sistem dalam memenuhi kebutuhan pengguna. Berikut penjelasan dimensi metode EUCS [25]:

1. Dimensi *Content*

   Dimensi *content* mengevaluasi sejauh mana pengguna merasa puas terhadap informasi atau konten yang disediakan oleh aplikasi. Kepuasan pengguna meningkat jika informasi yang disajikan lebih informatif dan relevan.

2. Dimensi *Accuracy*

   Dimensi *accuracy* mengukur tingkat ketepatan dan keakuratan data yang dihasilkan oleh sistem. Sistem yang semakin akurat ditunjukkan dengan semakin sedikitnya kesalahan yang terjadi.

3. Dimensi Format

   Dimensi format menilai tata letak, desain tampilan, dan estetika aplikasi. Faktor ini mencakup bagaimana tampilan memengaruhi kenyamanan pengguna.

4. Dimensi *Ease of Use*

   Dimensi *ease of use* mengukur kemudahan penggunaan aplikasi. Sistem yang mudah digunakan akan meningkatkan kenyamanan dan kepuasan pengguna.

5. Dimensi *Timeliness*

   Dimensi *timeliness* mengevaluasi kecepatan aplikasi dalam memberikan respons atau umpan balik kepada pengguna, yang merupakan indikator penting dalam mengukur efisiensi sistem.

2.7 Metode Penelitian Kuantitatif

Penelitian kuantitatif merupakan metode penelitian bertujuan untuk menguji teori atau hipotesis menggunakan data numerik dan analisis statistik [26]. Metode penelitian ini bersifat objektif, sistematis, dan terukur, dengan hasil yang dapat diaplikasikan pada populasi yang luas. Salah satu aspek penting dalam penelitian kuantitatif adalah teknik pengambilan sampel, yaitu proses memilih sebagian anggota dari populasi untuk dijadikan objek penelitian. Sugiyono [26] menjelaskan bajwa terdapat beberapa teknik pengambilan sampel yang terbagi menjadi dua jenis, yakni *Probability Sampling* dan *Non-Probability Sampling*. Teknik pengambilan sampel ini dipilih tergantung pada tujuan penelitian, ketersediaan data, dan sumber daya yang dimiliki [26].

Penelitian ini menggunakan metode pengambilan sampel *purposive sampling*. Teknik ini termasuk ke dalam jenis *non-random sampling*, di mana peneliti memilih sampel berdasarkan kriteria atau karakteristik tertentu yang sesuai dengan tujuan penelitian [27]. Dengan demikian, pemilihan sampel dilakukan secara selektif untuk memastikan data yang dikumpulkan relevan dengan masalah penelitian. *Purposive sampling* memungkinkan peneliti untuk menentukan sampel berdasarkan pertimbangan spesifik yang telah ditetapkan sebelumnya [28].

Roscoe, dalam bukunya *Research Methods for Business*, memberikan panduan mengenai ukuran sampel yang ideal, yaitu berkisar antara 30 hingga 500 sampel untuk penelitian [29]. Berdasarkan panduan tersebut, penelitian ini menggunakan 30 responden sebagai jumlah sampel penelitian.

2.7.1 Skala Likert

Skala Likert adalah alat ukur yang dipakai dalam penelitian untuk menilai berdasarkan persepsi responden. Skala ini memiliki kategori dan skor jawaban yang akan dipilih sesuai dengan pandangan atau persepsi responden. Adapun kategori dan skor yang digunakan dalam Skala Likert dapat dilihat pada Tabel 2.2 [30].

Tabel 2.2. Skala likert

Skor	Skala
5	Sangat Setuju
4	Setuju
3	Cukup Setuju
2	Tidak Setuju
1	Sangat Tidak Setuju

2.7.2 Analisis Tabel Deskriptif

Data yang dianalisis berasal dari data primer yang diperoleh dari kuisioner yang telah dibagikan kepada pengguna *website*. Proses analisis data kemudian dilakukan dengan metode deskriptif, yakni dengan mengumpulkan data dari kuisioner yang telah diisi oleh pengguna, yang kemudian dihitung nilai *mean*-nya, lalu diubah menjadi bentuk persentase untuk dikategorikan. Perhitungan nilai *mean* dilakukan untuk menilai tingkat kepuasan pengguna terhadap *website*. Rumus yang digunakan dalam perhitungan tersebut dapat dilihat pada persamaan Rumus 2.5 [30].

$$X = \frac{x_1 + x_2 + x_3 + ... + x_n}{N} \tag{2.5}$$

Keterangan:

X = perhitungan *mean*

$x_1 + x_2 + x_3 + ... + x_n$ = jumlah semua skor kuesioner

N = jumlah responden

Setelah mendapatkan nilai *mean*, maka dilakukan perubahan nilai *mean* ke dalam bentuk persentase untuk dapat dikategorikan. Proses perhitungan rentang nilai dapat dilihat pada persamaan Rumus 2.6.

$$Rentang\ Nilai = \frac{Nilai\ Mean}{Nilai\ Skala\ Tertinggi} \times 100 \quad (2.6)$$

Persentase yang diperoleh akan dikategorikan sesuai dengan tabel kategori tingkat kepuasan. Kategori tingkat kepuasan ini dapat dilihat pada tabel pada Tabel 2.3 [30].

Tabel 2.3. Kategori tingkat kepuasan

Persentase	Kategori
75.02 sampai $\leq$ 100	Sangat Tinggi
58.35 sampai $\leq$ 75.01	Tinggi
41.67 sampai $\leq$ 58.34	Kurang
25.00 sampai $\leq$ 41.66	Rendah
0 sampai $\leq$ 24.99	Sangat Rendah