

(Artikel Penelitian/Ulasan)

Implementasi Algoritma First In First Out (FIFO) Berbasis Web untuk Perhitungan dan Simulasi Berbagai Kasus Antrean

Akbar Alif Haikal¹, Zulfahmi Indra², Azril Arfansyah³, Ammar Kamil Al Abror⁴, Aldrik Bastio⁵

¹²³⁴⁵Ilmu Komputer, FMIPA, Universitas Negeri Medan Jalan Willem Iskandar, Pasar V Medan Estate, Percut Sei Tuan, Deli Serdang

¹akbaralifhaikal92@gmail.com, ²zulfahmi.indra@unimed.ac.id, ³azrilarfansyah403@gmail.com,
⁴ammarkamil1310@gmail.com ⁵aldrikbastio@gmail.com

* Penulis : Akbar Alif Haikal

Abstract: The *First In First Out* (FIFO) algorithm is a basic scheduling method widely used in queuing system prioritization. This study aims to implement the FIFO algorithm through a web-based application using HTML, CSS, and JavaScript programming languages. The developed system is designed to compute and simulate queueing scenarios based on *arrival time* and *burst time* parameters, producing outputs such as *start time*, *waiting time*, and *turn around time* automatically. The research methodology includes algorithm design, web interface development, and testing through several queueing case scenarios. Preliminary implementation results show that the system can display calculation outputs accurately in accordance with the FIFO theory. This application can serve as an interactive learning medium to help users understand queueing systems and the practical application of process scheduling algorithms.

Keywords: FIFO Algorithm; Queueing System; Web-Based Learning; Algorithm Visualization; Process Time Calculation.

Abstrak: Algoritma *First In First Out* (FIFO) merupakan metode penjadwalan sederhana yang secara luas digunakan dalam berbagai sistem antrean. Prinsip utama algoritma ini adalah entitas yang datang terlebih dahulu akan dilayani terlebih dahulu tanpa mempertimbangkan prioritas tertentu. Penelitian ini bertujuan untuk mengimplementasikan algoritma FIFO dalam bentuk aplikasi berbasis web menggunakan bahasa pemrograman HTML, CSS, dan JavaScript. Aplikasi ini dirancang untuk menghitung dan mensimulasikan sistem antrean berdasarkan parameter *arrival time* dan *burst time*, serta menghasilkan keluaran berupa *start time*, *waiting time*, dan *turn around time* secara otomatis. Metode penelitian meliputi tahap perancangan algoritma, pengembangan antarmuka web, serta pengujian melalui berbagai skenario antrean sederhana. Hasil implementasi awal menunjukkan bahwa sistem mampu menampilkan hasil perhitungan secara akurat sesuai teori dasar FIFO. Aplikasi ini dapat digunakan sebagai media pembelajaran interaktif yang membantu pengguna memahami konsep sistem antrean dan penerapan algoritma penjadwalan proses secara praktis.

Kata kunci: Algoritma FIFO; Sistem Antrean; Pembelajaran Berbasis Web; Visualisasi Algoritma; Perhitungan Waktu Proses.

Diterima: 13 Oktober 2025
Direvisi: 3 November 2025
Diterima: 4 November 2025
Diterbitkan: 5 November 2025
Versi sekarang: Januari 2026



Hak cipta: © 2025 oleh penulis.
Diserahkan untuk kemungkinan
publikasi akses terbuka berdasarkan
syarat dan ketentuan lisensi Creative
Commons Attribution (CC BY SA) (
<https://creativecommons.org/licenses/by-sa/4.0/>)

1. Pendahuluan

Antrean merupakan fenomena yang sangat umum terjadi, baik dalam sistem pelayanan publik maupun dalam sistem komputasi modern. Dalam konteks sistem operasi, konsep antrean digunakan untuk mengatur urutan eksekusi proses yang menunggu giliran untuk menggunakan sumber daya seperti CPU atau perangkat input/output (Putri, 2021). Salah satu pendekatan penjadwalan yang paling sederhana adalah **First In First Out (FIFO)**, yang bekerja berdasarkan urutan kedatangan proses (arrival time). Proses yang datang lebih awal akan diproses terlebih dahulu hingga selesai sebelum proses berikutnya dieksekusi (Silberschatz et al., 2020).

Dalam bidang industri, algoritma FIFO juga digunakan untuk pengelolaan stok barang agar barang yang lebih lama tersimpan digunakan lebih dulu, sehingga menghindari risiko kedaluwarsa (Sohrabi et al., 2021). Prinsip ini juga diterapkan pada sistem antrean pelayanan seperti rumah sakit, bank, dan layanan pelanggan (Attar et al., 2020). Kesederhanaan logika FIFO menjadikannya ideal untuk dijadikan dasar pembelajaran tentang teori antrean (*queueing theory*) dan manajemen proses dalam komputer.

Namun, pemahaman tentang hubungan antara *arrival time*, *burst time*, *waiting time*, dan *turn around time* sering kali sulit jika hanya dijelaskan secara teori. Oleh karena itu, diperlukan media pembelajaran berbasis teknologi yang mampu memperlihatkan bagaimana setiap proses diproses secara berurutan sesuai aturan FIFO. Aplikasi web interaktif menjadi solusi tepat karena dapat diakses dengan mudah, bersifat visual, dan tidak memerlukan instalasi perangkat lunak tambahan (Nishiyama et al., 2017).

Penelitian ini berfokus pada implementasi algoritma FIFO berbasis web yang mampu menghitung dan mensimulasikan berbagai skenario antrean sederhana. Sistem ini diharapkan dapat membantu pengguna, khususnya mahasiswa dan praktisi teknologi informasi, memahami prinsip dasar penjadwalan FIFO secara aplikatif dan mendalam.

2. Tinjauan Pustaka

2.1 Algoritma FIFO

Algoritma FIFO adalah metode penjadwalan non-preemptive yang memproses setiap entitas sesuai urutan kedatangannya (Kleinrock, 1975). Dalam konteks sistem operasi, algoritma ini dikenal juga sebagai First Come First Served (FCFS), di mana setiap proses dijadwalkan hingga selesai sebelum proses berikutnya dimulai (Stallings, 2018).

Penelitian oleh Parinduri & Hutagalung (2019) menunjukkan bahwa meskipun algoritma FIFO sederhana, efisiensinya dapat berkurang ketika variasi waktu proses sangat tinggi, karena proses yang panjang dapat menyebabkan keterlambatan signifikan bagi proses lain.

2.2 Sistem Antrean

Sistem antrean (*queueing system*) merupakan model matematis yang menggambarkan proses kedatangan entitas, proses pelayanan, dan disiplin layanan (Gross et al., 2023). Model paling dasar adalah M/M/1, di mana kedatangan mengikuti distribusi Poisson dan waktu pelayanan mengikuti distribusi eksponensial. Sistem ini memiliki parameter utama berupa laju kedatangan (λ) dan laju pelayanan (μ). Berdasarkan teori Little's Law, hubungan antara jumlah rata-rata pelanggan dalam sistem (L), waktu rata-rata dalam sistem (W), dan laju kedatangan (λ) dapat dinyatakan sebagai:

$$L = \lambda W$$

Model FIFO digunakan sebagai basis untuk sistem antrean tanpa prioritas dan memberikan pemahaman dasar mengenai perilaku sistem yang stabil (Cooper, 1990).

Dalam konteks komputasi modern, teori antrean juga diaplikasikan pada pengelolaan proses CPU, trafik jaringan, hingga sistem transaksi daring (Raicu, 2023). Dengan demikian,

pemahaman mengenai sistem antrean berbasis FIFO menjadi fondasi penting dalam analisis kinerja sistem komputasi maupun industri.

2.3 Pembelajaran Berbasis Web

Media pembelajaran berbasis web memanfaatkan teknologi internet untuk menyampaikan materi pembelajaran secara interaktif dan fleksibel (Alkhazl et al., 2017). Dalam konteks algoritma penjadwalan, visualisasi berbasis web memungkinkan pengguna untuk memahami proses secara langsung melalui simulasi yang interaktif (Jungmann, 2021). Pembelajaran berbasis web juga memberikan keunggulan dalam hal ketersediaan, karena dapat diakses kapan saja dan di mana saja tanpa batas perangkat.

Penelitian Nishiyama, Kagawa, dan Imai (2017) menunjukkan bahwa penggunaan visualisasi berbasis web pada algoritma penjadwalan meningkatkan tingkat pemahaman mahasiswa hingga 40% dibandingkan pembelajaran konvensional berbasis teks. Dengan demikian, penerapan aplikasi FIFO berbasis web dalam penelitian ini diharapkan dapat meningkatkan efektivitas pembelajaran di bidang sistem operasi dan teori antrean.

2.4 Visualisasi Algoritma

Visualisasi algoritma adalah metode representasi visual untuk membantu pengguna memahami proses komputasi (Alkhazl et al., 2017). Dalam konteks penjadwalan, visualisasi memungkinkan pengguna untuk melihat urutan eksekusi proses secara dinamis, termasuk parameter seperti arrival time, start time, dan turn around time. Menurut penelitian oleh Formosa Publisher (2023), implementasi visualisasi FIFO pada jaringan komputer membantu pengguna memahami urutan pemrosesan paket secara intuitif dan akurat.

Visualisasi yang baik tidak hanya menampilkan hasil akhir perhitungan, tetapi juga menggambarkan alur logika dan urutan proses, sehingga pembelajaran menjadi lebih bermakna (Raicu, 2023).

2.5 Penelitian Terkait

Beberapa penelitian sebelumnya telah mengembangkan aplikasi simulasi algoritma penjadwalan CPU berbasis web, seperti Round Robin dan Shortest Job First (Putra & Purnomo, 2022; Belferik & Banjarnahor, 2025). Namun, penelitian tentang visualisasi algoritma FIFO berbasis web masih terbatas. Studi oleh Widiarto et al. (2023) menunjukkan bahwa penggunaan antarmuka sederhana berbasis HTML dan JavaScript dapat meningkatkan keterlibatan pengguna dalam memahami konsep penjadwalan. Oleh karena itu, penelitian ini berkontribusi dengan menghadirkan sistem simulasi FIFO yang berfokus pada kesederhanaan, keakuratan perhitungan, dan kemudahan penggunaan.

3. Metode Penelitian

Penelitian ini menggunakan pendekatan pengembangan sistem terapan (applied system development) dengan tujuan menghasilkan aplikasi web interaktif untuk simulasi dan perhitungan algoritma FIFO. Metode penelitian terdiri dari empat tahap utama: desain sistem, perancangan algoritma, implementasi berbasis web, dan pengujian sistem.

3.1 Desain Sistem

Sistem dirancang sebagai aplikasi berbasis web yang berjalan sepenuhnya di sisi klien (*client-side*). Struktur utamanya terdiri atas:

- Form input data – pengguna memasukkan *arrival time* dan *burst time* untuk setiap proses.
- Mesin perhitungan algoritma FIFO – sistem menghitung *start time*, *waiting time*, dan *turn around time* berdasarkan urutan kedatangan.
- Tabel hasil simulasi – menampilkan hasil perhitungan secara otomatis dalam format tabel yang mudah dibaca.
- Visualisasi urutan proses – diagram waktu eksekusi ditampilkan untuk membantu pemahaman alur antrean.

Arsitektur ini memastikan sistem ringan, interaktif, dan dapat dijalankan di seluruh peramban modern tanpa memerlukan instalasi tambahan (Jungmann, 2021).

3.2 Perancangan Algoritma FIFO

Logika utama perhitungan mengikuti langkah-langkah berikut:

- Urutkan proses berdasarkan *arrival time*.
- Tentukan waktu mulai (*start time*) proses pertama sama dengan *arrival time*-nya.
- Hitung proses berikutnya menggunakan

$$\text{Start Time} = \max(\text{Arrival Time}, \text{Completion Time})$$
- Hitung waktu tunggu dan penyelesaian dengan rumus yang sudah ada yaitu

$$\text{Waiting time} = \text{Start Time} - \text{Arrival Time}, \text{Turn Around Time} = \text{Waiting Time} + \text{Burst Time}$$
- Simpan hasil dalam tabel keluaran.

Algoritma ini diimplementasikan menggunakan struktur array dan perulangan *for* dalam JavaScript agar efisien dan mudah dimengerti.

3.3 Implementasi Berbasis Web

Implementasi sistem menggunakan:

- HTML untuk membangun struktur form input dan tabel hasil,
- CSS untuk mempercantik antarmuka dan mengatur tata letak,
- JavaScript untuk melakukan perhitungan otomatis berdasarkan rumus FIFO.

Perhitungan dijalankan di browser secara real time setelah pengguna menekan tombol “Hitung”. Hasil perhitungan akan langsung muncul dalam tabel di halaman yang sama tanpa perlu memuat ulang halaman. Prinsip desain antarmuka mengutamakan kesederhanaan, responsivitas, dan kemudahan penggunaan (Nishiyama et al., 2017).

3.4 Pengujian Sistem

Tahap pengujian dilakukan dengan metode *black-box testing* untuk memverifikasi kebenaran hasil perhitungan. Pengujian melibatkan beberapa skenario:

1. Semua proses datang bersamaan (*arrival time* sama).
2. Proses datang bertahap dengan waktu kedatangan berbeda.

3. Variasi *burst time* untuk menguji pengaruhnya terhadap waktu tunggu dan penyelesaian.

4. Hasil dan Pembahasan

Aplikasi simulasi **First In First Out (FIFO)** berbasis web yang dikembangkan pada penelitian ini dirancang untuk menghitung serta memvisualisasikan proses penjadwalan CPU berdasarkan urutan kedatangan proses. Sistem secara otomatis menampilkan hasil perhitungan parameter seperti *Start Time*, *Completion Time*, *Turn Around Time*, dan *Waiting Time*, serta menyajikannya dalam bentuk *Gantt Chart* agar pengguna dapat memahami alur eksekusi secara visual. Aplikasi ini dikembangkan menggunakan bahasa pemrograman **HTML**, **CSS**, dan **JavaScript**, dan dijalankan sepenuhnya di sisi klien (*client-side*). Dengan demikian, sistem dapat digunakan secara langsung melalui peramban (*browser*) tanpa memerlukan instalasi tambahan, menjadikannya ringan, interaktif, dan mudah diakses di berbagai perangkat.

1. Tampilan Program

Antarmuka aplikasi didesain dengan gaya modern yang memadukan warna biru dan kuning agar tampilan terlihat menarik namun tetap profesional. Struktur halaman utama dibagi menjadi empat bagian utama, yaitu:

1. **Header**, berisi judul “*Sistem Penjadwalan FCFS (FIFO)*” dan deskripsi singkat mengenai algoritma yang digunakan.
2. **Bagian Input Data**, yang memungkinkan pengguna menentukan jumlah proses serta memasukkan nilai *Arrival Time (AT)* dan *Burst Time (BT)* dalam tabel interaktif.
3. **Bagian Hasil Penjadwalan**, yang menampilkan tabel hasil perhitungan otomatis beserta *Gantt Chart* berwarna untuk memvisualisasikan urutan proses.
4. **Bagian Langkah Perhitungan Manual**, yang menjelaskan rumus dan tahapan perhitungan setiap proses secara rinci agar mudah dipahami.
5. Selain itu, sistem juga menampilkan nilai rata-rata *Waiting Time* dan *Turn Around Time* untuk memberikan gambaran umum kinerja keseluruhan algoritma

Sistem Penjadwalan FCFS (FIFO)
Algoritma First Come, First Served / First In, First Out

Konsep Dasar Penjadwalan

Salah	Penjelasan	Rumus
Arrival Time (AT)	Waktu sebuah proses masuk ke antrian siap.	Input Pengguna
Burst Time (BT)	Waktu yang dibutuhkan CPU untuk menyelesaikan eksekusi proses.	Input Pengguna
Completion Time (CT)	Waktu ketika proses selesai dieksekusi.	Start Time + BT
Turn Around Time (TAT)	Total waktu yang dibutuhkan proses di sistem (dari datang hingga selesai).	CT - AT
Waiting Time (WT)	Waktu yang dibutuhkan proses untuk menunggu di antrian sebelum dieksekusi.	TAT - BT

1. Masukkan Data Proses

Jumlah Proses:

Proses	Arrival Time (AT)	Burst Time (BT)
P1	2	5
P2	3	3
P3	6	6

Hitung Penjadwalan

Gambar 1

2. Cara Penggunaan

Berikut merupakan Langkah-langkah penggunaan aplikasi cukup sederhana dan intuitif:

1. Pengguna membuka halaman aplikasi melalui peramban web.
2. Menentukan jumlah proses yang akan diuji (misalnya 3 proses).
3. Mengisi nilai *Arrival Time* dan *Burst Time* untuk masing-masing proses.
4. Menekan tombol “**Hitung Penjadwalan**” untuk memulai simulasi.
5. Sistem kemudian akan:
6. Mengurutkan proses berdasarkan waktu kedatangan (*arrival time*).
7. Menghitung seluruh parameter penjadwalan.
8. Menampilkan hasil dalam bentuk tabel serta *Gantt Chart* secara otomatis.
9. Semua hasil dapat langsung diamati tanpa perlu memuat ulang halaman, karena proses perhitungan berlangsung secara *real-time* di dalam peramban.

The screenshot shows a web form titled "Masukkan Data Proses". It contains a table with three columns: "Proses", "(TA) arrival time", and "(BT) burst time". The table has three rows for processes P1, P2, and P3. Each row has input fields for the arrival time and burst time. Below the table is a dropdown menu labeled "Pilih Proses".

Gambar 2

3. Studi Kasus

Sebagai contoh, digunakan tiga proses dengan parameter berikut:

Proses	Arrival Time (AT)	Burst Time (BT)
P1	0	5
P2	0	3
P3	0	8

Ketiga proses tiba pada waktu yang sama, sehingga sistem mengeksekusi proses sesuai urutan kedatangan: P1 → P2 → P3. Berdasarkan aturan FIFO, hasil perhitungan diperoleh sebagai berikut

Proses	AT	BT	Start Time	Completion Time	Turn Around Time	Waiting Time
P1	0	5	0	5	5	0
P2	0	3	5	8	8	5
P3	0	8	8	16	16	8

Dari hasil tersebut diperoleh:

Rata-rata **Turn Around Time (TAT)** = $(5 + 8 + 16) / 3 = 9,67$

Rata-rata **Waiting Time (WT)** = $(0 + 5 + 8) / 3 = 4,33$

4. Eksekusi Program

Seluruh proses perhitungan dijalankan di sisi klien menggunakan fungsi utama `calculateFCFS()` pada berkas `script.js`. Fungsi ini bertanggung jawab untuk:

1. Mengambil data input *Arrival Time* dan *Burst Time* dari pengguna.
2. Mengurutkan proses berdasarkan waktu kedatangan.

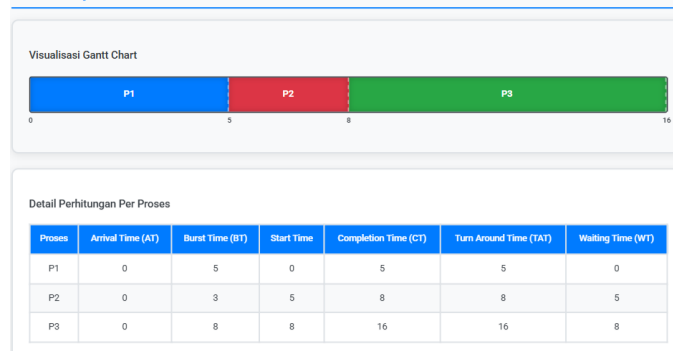
Melakukan perhitungan dengan rumus:

- $Start\ Time = \max(Completion\ Time\ sebelumnya, Arrival\ Time)$
- $Completion\ Time\ (CT) = Start\ Time + Burst\ Time$
- $Turn\ Around\ Time\ (TAT) = CT - AT$
- $Waiting\ Time\ (WT) = TAT - BT$

3. Menampilkan hasil dalam tabel dan *Gantt Chart* secara otomatis.

Selain itu, sistem juga menampilkan langkah-langkah perhitungan manual untuk setiap proses sehingga pengguna dapat melakukan verifikasi hasil dengan mudah.

2. Hasil Penjadwalan Proses



Gambar 3

5. Alur dan Cara Kerja Sistem

Secara umum, alur kerja sistem dapat dijelaskan melalui tujuh tahapan berikut:

1. **Input Data Proses** – pengguna mengisi nilai *Arrival Time* dan *Burst Time*.
2. **Validasi Input** – sistem memastikan data yang dimasukkan valid dan positif.
3. **Sortasi Proses** – proses diurutkan berdasarkan waktu kedatangan.
4. **Perhitungan Eksekusi** – sistem menghitung *Start Time*, *Completion Time*, *TAT*, dan *WT*.
5. **Visualisasi Gantt Chart** – sistem menampilkan hasil dalam bentuk grafik batang berwarna sesuai durasi proses.
6. **Perhitungan Rata-rata** – sistem menghitung nilai rata-rata *TAT* dan *WT* dari seluruh proses.
7. **Tampilan Akhir** – hasil akhir berupa tabel, visualisasi, dan langkah-langkah perhitungan ditampilkan secara utuh pada halaman yang sama.

6. Hasil

Berdasarkan serangkaian pengujian terhadap beberapa skenario, aplikasi menunjukkan hasil yang akurat dan stabil.

1. Pada kondisi di mana semua proses memiliki waktu kedatangan yang sama, urutan eksekusi mengikuti urutan input tanpa adanya *idle time*.
2. Pada kondisi dengan waktu kedatangan yang berbeda, sistem secara otomatis menyesuaikan urutan eksekusi sesuai dengan *arrival time* masing-masing proses.
3. Hasil rata-rata *Waiting Time* dan *Turn Around Time* yang dihasilkan sesuai dengan teori dasar algoritma FIFO dan dapat dibuktikan dengan perhitungan manual.

Langkah Perhitungan Manual

A. Perhitungan Detail Setiap Proses

1. Proses P1 (AT=0, BT=5)

1. Start Time (Waktu Mulai):
Rumus: Maksimum(CT Proses Sebelumnya, AT Proses Saat Ini)
Hitung: $\text{Maksimum}(0, 0) = 0$

2. Completion Time (CT):
Rumus: Start Time + Burst Time (BT)
Hitung: $0 + 5 = 5$

3. Turn Around Time (TAT):
Rumus: Completion Time (CT) - Arrival Time (AT)
Hitung: $5 - 0 = 5$

4. Waiting Time (WT):
Rumus: Turn Around Time (TAT) - Burst Time (BT)
Hitung: $5 - 5 = 0$

2. Proses P2 (AT=0, BT=3)

1. Start Time (Waktu Mulai):
Rumus: Maksimum(CT Proses Sebelumnya, AT Proses Saat Ini)
Hitung: $\text{Maksimum}(5, 0) = 5$

2. Completion Time (CT):
Rumus: Start Time + Burst Time (BT)
Hitung: $5 + 3 = 8$

3. Turn Around Time (TAT):
Rumus: Completion Time (CT) - Arrival Time (AT)
Hitung: $8 - 0 = 8$

4. Waiting Time (WT):
Rumus: Turn Around Time (TAT) - Burst Time (BT)
Hitung: $8 - 3 = 5$

3. Proses P3 (AT=0, BT=8)

1. Start Time (Waktu Mulai):
Rumus: Maksimum(CT Proses Sebelumnya, AT Proses Saat Ini)
Hitung: $\text{Maksimum}(8, 0) = 8$

2. Completion Time (CT):
Rumus: Start Time + Burst Time (BT)
Hitung: $8 + 8 = 16$

3. Turn Around Time (TAT):
Rumus: Completion Time (CT) - Arrival Time (AT)
Hitung: $16 - 0 = 16$

4. Waiting Time (WT):
Rumus: Turn Around Time (TAT) - Burst Time (BT)
Hitung: $16 - 8 = 8$

Gambar 4

5. Kesimpulan

Penelitian ini telah berhasil merancang dan mengimplementasikan **aplikasi simulasi algoritma First In First Out (FIFO)** berbasis web sebagai media pembelajaran interaktif untuk memahami konsep penjadwalan proses pada sistem operasi. Aplikasi yang dibangun menggunakan **HTML, CSS, dan JavaScript** ini mampu menghitung parameter penjadwalan seperti Start Time, Completion Time, Turn Around Time, dan Waiting Time secara otomatis. Selain itu, sistem juga menampilkan visualisasi dalam bentuk Gantt Chart yang memperlihatkan urutan eksekusi proses secara jelas dan mudah dipahami. Hasil pengujian menunjukkan bahwa aplikasi bekerja dengan baik dan menghasilkan perhitungan yang akurat. Ketika seluruh proses memiliki waktu kedatangan yang sama, sistem mengeksekusi proses sesuai urutan input tanpa waktu menganggur (*idle time*). Sementara itu, pada kondisi di mana waktu kedatangan berbeda, sistem dapat menyesuaikan urutan eksekusi berdasarkan prinsip dasar algoritma FIFO, yaitu proses yang datang lebih awal akan diproses terlebih dahulu. Secara keseluruhan, aplikasi yang dikembangkan telah memenuhi tujuan penelitian, yakni menyediakan **media pembelajaran berbasis web yang ringan, interaktif, dan mudah digunakan** untuk memperdalam pemahaman mengenai konsep dasar penjadwalan CPU. Visualisasi yang dihasilkan tidak hanya mempermudah proses belajar, tetapi juga membantu pengguna melihat hubungan logis antara *arrival time*, *burst time*, *waiting time*, dan *turn around time* secara nyata dan terstruktur.

References

- Alkhazl, F., Hasan, M., & Ibrahim, M. (2017). *Applying visualization for teaching scheduling algorithms*. California State University ScholarWorks.
- Attar, H., Hashim, S., & Diko, I. (2020). *Review and performance evaluation of FIFO, PQ, CQ, FQ, WFQ, and CBWFQ queue scheduling mechanisms*. *International Journal of Engineering Research and Technology*, 9(3), 112–118.
- Belferik, R., & Banjarnahor, E. (2025). *Analisis average waiting time penjadwalan CPU menggunakan algoritma shortest remaining first dan algoritma round robin*. *JDMIS: Journal of Data Mining and Information Systems*, 3(1), 43–53.
- Cooper, R. B. (1990). *Queueing theory*. In *Handbooks in Operations Research and Management Science* (Vol. 1). Elsevier.
- Formosa Publisher. (2023). *Analysis of first in first out (FIFO) bandwidth packet queuing*. *East Asian Journal of Multidisciplinary Research*, 2(10), 4567–4578.
- Gross, D., Shortle, J. F., Thompson, J. M., & Harris, C. M. (2023). *Fundamentals of queueing theory* (6th ed.). Wiley.
- Jungmann, T. (2021). *Implementation of a web-based exercise platform for algorithm visualization*. Technical University of Munich.
- Kleinrock, L. (1975). *Queueing systems, volume I: Theory*. Wiley-Interscience.
- Motamedi, Z., Baygi, S. F., & Rezaei, M. (2020). *Queueing theory application in food supply chain: FIFO method implementation*. *Journal of Quality and Productivity Optimization*, 3(2), 45–54.
- Nishiyama, K., Kagawa, K., & Imai, Y. (2017). *Development of a CPU scheduling algorithms visualizer using JavaScript*. *IEEE Transactions on Electronics, Information and Systems*, 137(12), 1641–1648.
- Parinduri, I., & Hutagalung, S. N. (2019). *Teknik penjadwalan prosesor FIFO, SJF non-preemptive, round robin*. *Prosiding Seminar Nasional Riset Information Science (SENARIS)*.
- Putra, T. D., & Purnomo, R. (2022). *Simulation of priority round robin scheduling algorithm*. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 7(4), 2170–2181.
- Putri, R. A. (2021). *Aplikasi simulasi algoritma penjadwalan sistem operasi*. *Jurnal Teknologi Informasi*, 5(1), 98–102.
- Raicu, S. (2023). *Effects of the queue discipline on system performance*. *Systems*, 3(1), 3.
- Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). *Operating system concepts* (10th ed.). Wiley.
- Sohrabi, M., Baygi, S. F., & Rezaei, M. (2021). *A simple empirical inventory model using FIFO for managing material expiry*. *BMC Health Services Research*, 21(5), 145–157.

Stallings, W. (2018). *Operating systems: Internals and design principles* (9th ed.). Pearson.

The Effect of Queuing Mechanisms: FIFO, PQ, and WFQ on Network Routers. (2021). *Scientific Research Publishing*.

Widiarto, W., Maheswari, D., & Arianto, K. J. (2023). *Implementasi algoritma round robin dan priority pada sistem antrian rumah sakit*. *Jurnal Teknologi Informasi dan Sistem Komputer*, 11(2), 77–84.

Research on the Queuing Theory Based on M/M/1 Queuing Model. (2022). *ResearchGate*.