Seminar Nasional Sains Data 2025 (SENADA 2025)
UPN "Veteran" Jawa Timur
E-ISSN 2808-5841
P-ISSN 2808-7283

Penjadwalan Shift Kerja Berbasis Permintaan Karyawan dengan Algoritma Simulated Annealing

Rika Pramudhita[1], Intan Dzikria[2]

[1]*Program Studi Teknik Informatika, Universitas 17 Agustus 1945 Surabaya*
[2]*Program Studi Sistem dan Teknologi Informasi, Universitas 17 Agustus 1945 Surabaya*
Corresponding author email: intandzikria@untag-sby.ac.id

Abstract: *Shift scheduling is one of the main challenges faced by manufacturing companies that operate 24 hours a day. Suboptimal scheduling can affect operational efficiency and employee welfare. Therefore, this study aims to develop an employee shift scheduling system based on employee demand using the Simulated Annealing algorithm. This algorithm was chosen because of its ability to solve optimization problems efficiently, including in complex work scheduling. The designed system is also equipped with a flexibility feature that allows employees to submit requests for shift changes, leave, or holidays more easily. The system development follows the Waterfall method. The results of the study show that the SA algorithm is able to create employee shift schedules with predetermined limitations and based on demand. The results of this study are expected to provide efficient, fair, and adaptive scheduling solutions to employee and company needs.*
Keywords: *simulated annealing algorithm, scheduling, work scheduling, information system*

Abstrak: Penjadwalan *shift* kerja merupakan salah satu tantangan utama yang dihadapi oleh perusahaan manufaktur yang beroperasi 24 jam sehari. Penjadwalan yang tidak optimal dapat memengaruhi efisiensi operasional dan kesejahteraan karyawan. Oleh karena itu, penelitian ini bertujuan untuk mengembangkan sistem penjadwalan *shift* kerja karyawan berbasiskan pada permintaan karyawan menggunakan algoritma *Simulated Annealing*. Algoritma ini dipilih karena kemampuannya dalam menyelesaikan permasalahan optimasi dengan efisien, termasuk dalam penjadwalan kerja yang kompleks. Sistem yang dirancang juga dilengkapi dengan fitur fleksibilitas yang memungkinkan karyawan untuk mengajukan permohonan (*request*) pergantian *shift*, cuti, atau libur dengan lebih mudah. Pengembangan sistem mengikuti metode *Waterfall*. Hasil penelitian menunjukkan bahwa algoritma SA mampu membuat jadwal *shift* kerja karyawan dengan batasan-batasan yang telah ditentukan dan berdasarkan permintaan. Hasil penelitian ini diharapkan dapat memberikan solusi penjadwalan yang efisien, adil, dan adaptif terhadap kebutuhan karyawan dan perusahaan.
Kata kunci: algoritma *simulated annealing*, penjadwalan, *shift* kerja, sistem informasi

I. PENDAHULUAN

Sebagian besar perusahaan bergantung pada sistem informasi yang terintegrasi untuk meningkatkan efisien dan efektivitas operasional. Implementasi teknologi informasi dalam berbagai aspek operasional perusahaan, seperti manajemen rantai pasokan dan pengelolaan persediaan, telah berhasil meningkatkan efisiensi, menurunkan biaya, dan meningkatkan kualitas produk [1]. Salah satu implementasi teknologi informasi juga pada aspek sumber daya manusia terkait penjadwalan kerja karyawan, terutama pada perusahaan yang menerapkan sistem kerja *shift*. Pada perusahaan yang beroperasi dengan sistem *shift*, seperti industri manufaktur, layanan kesehatan, pelayanan publik, dan sektor retail, penjadwalan *shift* menjadi elemen kunci untuk menjaga kontinuitas operasional dan memastikan ketersediaan tenaga kerja sepanjang waktu.

Shift kerja adalah pembagian waktu kerja dalam satu hari yang memungkinkan pekerja bergiliran bekerja, sehingga operasional dapat terus berlangsung tanpa henti [2]. Namun, penyusunan jadwal *shift* secara manual masih menjadi tantangan besar, terutama bagi perusahaan yang beroperasi selama 24 jam [3]. Proses ini memerlukan ketelitian tinggi karena kesalahan seperti menempatkan *shift* malam yang langsung diikuti oleh *shift* pagi dapat menyebabkan kelelahan dan penurunan kinerja karyawan [2]. Ketidakadilan dalam pembagian shift juga dapat menimbulkan konflik internal, khususnya jika terdapat karyawan yang terus menerus mendapat *shift* tertentu [2].

Metode penyusunan jadwal *shift* secara tradisional, seperti menggunakan *Microsoft Excel*, masih umum digunakan [3]. Meskipun familiar, pendekatan ini cenderung memakan waktu dan rentan

terhadap bentrok jadwal, menghabiskan banyak waktu, dan menciptakan masalah operasional yang berkelanjutan [3]. Sedangkan pengelolaan jadwal yang efektif sangat penting dalam menjaga produktivitas serta kesehatan karyawan, karena menentukan waktu kerja, istirahat, dan libur secara seimbang [4].

Untuk menjawab tantangan ini, diperlukan solusi berupa sistem terkomputerisasi yang dapat mengurangi *human error* dan mempercepat proses penyusunan jadwal [2]. Salah satu pendekatan yang potensial adalah penggunaan algoritma *Simulated Annealing*, sebuah metode heuristik yang efektif untuk mencari solusi optimal dalam berbagai masalah optimasi [5]. Algoritma ini bekerja dengan mensimulasikan proses pendinginan logam dan mampu keluar dari solusi lokal yang tidak optimal [5].

Algoritma SA dipilih dalam penelitian ini karena memiliki kemampuan untuk menghindari jebakan pada solusi optimal lokal, sehingga lebih handal dibandingkan metode heuristik lainnya dalam konteks masalah optimasi yang kompleks [6]. Selain itu, algoritma ini juga dikenal mampu menghasilkan solusi dalam waktu yang relatif cepat [7]. Dengan karakteristik tersebut, SA sangat sesuai untuk diterapkan pada sistem penjadwalan *shift* yang memerlukan pencarian solusi optimal dalam ruang pencarian yang luas dan melibatkan berbagai constraint penjadwalan yang saling berinteraksi.

Algoritma SA sebelumnya telah diterapkan dalam konteks penjadwalan produksi, seperti untuk meminimalkan makespan [6], serta dalam proses produksi industri dupa [5]. Dibandingkan dengan algoritma lain seperti *Genetic Algorithm* (GA), yang telah banyak digunakan dalam penjadwalan shift kerja [8] [3], SA memiliki struktur yang lebih sederhana dan parameter kontrol yang lebih sedikit.

GA memiliki beberapa kelemahan, salah satunya adalah tidak selalu menjamin pencapaian solusi *optimum global* dan membutuhkan evaluasi dalam jumlah besar terhadap fungsi kesesuaian (*fitness function*) [9]. Selain itu, penerapan algoritma GA tergolong kompleks dan sulit, serta membutuhkan pemahaman dan implementasi teknis yang cukup rumit [10]. Sebaliknya, SA bekerja dengan pendekatan iteratif berbasis satu solusi yang dimodifikasi secara bertahap dan dievaluasi berdasarkan fungsi objektif. Keunggulan utama SA dibandingkan metode lainnya terletak pada kemampuannya untuk menghindari jebakan optimal lokal melalui mekanisme penerimaan solusi yang kurang baik dengan probabilitas tertentu, yang dikendalikan oleh parameter suhu [6]. Selain itu, algoritma SA mampu menemukan solusi dalam waktu yang relatif cepat, menjadikannya cocok untuk permasalahan penjadwalan yang memerlukan efisiensi proses [7].

Namun, penelitian sebelumnya belum mengembangkan algoritma SA dalam konteks penjadwalan *shift* kerja karyawan, terutama dalam sistem yang memberikan fleksibilitas kepada karyawan untuk mengajukan permohonan (*request*) pergantian *shift*, cuti, atau libur pada hari tertentu.

Penelitian ini bertujuan untuk mengembangkan sistem penjadwalan *shift* karyawan berbasiskan *request* atau permintaan yang menggunakan algoritma SA. Sistem ini diharapkan dapat mengatasi tantangan penjadwalan yang sering dihadapi perusahaan dan memberikan fleksibilitas pada karyawan untuk melakukan permintaan penjadwalan pada hari tertentu. Dengan adanya sistem ini, perusahaan dapat meningkatkan efektivitas operasional, mengurangi potensi kesalahan manusia, dan memberikan fleksibilitas lebih besar kepada karyawan dalam mengelola jadwal kerja mereka.

II. TINJAUAN PUSTAKA

1. Penjadwalan Shift Kerja

Penjadwalan merupakan proses penempatan sumber daya dalam rentang waktu tertentu untuk mengatur sumber daya sesuai dengan urutan pekerjaan yang telah ditentukan [11]. Tujuan dari penjadwalan adalah untuk mencapai efisiensi dan efektivitas dalam pelaksanaan aktivitas, terutama dalam pengelolaan tenaga kerja di suatu organisasi. [12] menjelaskan bahwa penjadwalan *shift* kerja

adalah pembagian waktu selama 24 jam, di mana satu atau sekelompok pekerja diatur untuk bekerja setelah kelompok sebelumnya menyelesaikan tugasnya.

Menurut [13], terdapat dua jenis sistem *shift* kerja, yaitu (1) *shift* tetap, di mana karyawan selalu bekerja pada periode waktu yang sama dan (2) *shift* rotasi, di mana jadwal kerja karyawan berubah-ubah secara berkala, misalnya dari *shift* pagi ke *shift* malam. Sistem kerja *shift* umumnya dibagi menjadi tiga periode: pagi (08.00–16.00), sore (16.00–24.00), dan malam (00.00–08.00), masing-masing mencakup waktu istirahat [13]. Pembagian waktu ini bertujuan untuk mempertahankan keseimbangan antara kebutuhan operasional dan kesejahteraan karyawan.

Maurits dan Widodo, dikutip dalam [13], menekankan beberapa prinsip penting dalam penyusunan jadwal *shift*, yaitu (1) pola rotasi maju dengan pergantian *shift* tidak lebih dari dua minggu, dengan libur 2 hari per minggu; (2) per durasi *shift* adalah 8 jam dan apabila lebih maka diperlukan penyesuaian beban kerja; (3) untuk pekerja *shift* malam dianjurkan istirahat siang dan pekerjaan dengan tingkat kritis tinggi dilakukan sebelum pukul 4 pagi; serta (4) memperhitungkan faktor demografi seperti jenis kelamin dan usia.

2. Algoritma Simulated Annealing

Algoritma *Simulated Annealing* (SA) terinspirasi dari proses *annealing* (pendinginan) yang kemudian diimplementasikan untuk menangani masalah optimasi oleh Kirkpatrick dan rekan-rekannya pada tahun 1983 [14]. Tujuan utama dari Algoritma SA adalah untuk meminimalkan fungsi objektif, di mana algoritma ini melakukan peningkatan secara bertahap dengan memperbaiki solusi yang dihasilkan oleh teknik-teknik penjadwalan heuristik [14]. SA mengatasi jebakan lokal minimum dengan probabilitas tertentu untuk menerima solusi yang lebih buruk, memungkinkan eksplorasi lebih luas. Fungsi *Boltzman* digunakan untuk menerima solusi probabilitas awal, menggunakan Rumus (1) Rumus probabilitas untuk menerima solusi yang lebih buruk adalah dengan fungsi *Boltzmann* [15], dimana E merupakan perbedaan antara solusi baru atau solusi saat ini, dan T merupakan temperatur atau suhu saat ini. Rumus (2) kemudian digunakan untuk melakukan pembaruan nilai parameter temperatur berdasarkan penurunan suhu α atau yang dikenal dengan *annealing schedule* [16], dimana α sebagai konstanta dengan nilai kurang dari 1 untuk menurunkan parameter kontrol.

$$P(\Delta E) = e^{\frac{-\Delta E}{T}} \qquad (1)$$

$$T = \alpha \times T \qquad (2)$$

Untuk memperjelas penerapan pada persamaan SA yang dijelaskan sebelumnya, berikut ini disajikan pseudocode yang menggambarkan langkah-langkah dalam menghitung SA untuk mencari solusi optimal dari masalah optimasi. *Pseudocode* ini menggambarkan bahwa SA dimulai dengan solusi awal s yang akan diperbarui sering berjalannya iterasi hingga mencapai batas maksimum k_{max} dan juga memperbarui nilai suhu T untuk mengontrol penerimaan solusi yang kurang optimal.

Pseudocode 1 [11] : *Simulated Annealing*

Let $s = s_0$
For $k = 0$ through k_{max} (exclusive):
    $T \leftarrow$ temperature (k/k_{max})
    Pick a random neighbour, snew $\leftarrow$ neighbour(s)
    If $P(E(s), E(s_{new}), T) \geq$ random(0, 1), move to the new state:
        $s \leftarrow s_{new}$
Output: the final state s

SA memiliki keunggulan dalam menemukan solusi yang mendekati optimal dalam waktu relatif cepat dan fleksibel diterapkan pada berbagai jenis masalah optimasi. Namun, efektivitas algoritma sangat bergantung pada nilai parameter awal dan strategi pendinginan yang digunakan [7].

III. METODOLOGI

Penelitian ini bertujuan utama untuk mengembangkan sistem penjadwalan *shift* kerja karyawan berbasis algoritma SA yang mampu menghasilkan jadwal kerja optimal, sekaligus mengakomodasi permintaan (*request*) karyawan terhadap pergantian *shift*, cuti, dan hari libur. Untuk mendukung tercapainya pengembangan sistem penjadwalan *shift* karyawan menggunakan algoritma SA, penelitian ini mengumpulkan data awal melalui studi literatur dan observasi. Berbagai literatur akademis digunakan untuk memahami algoritma SA dan studi kasus penjadwalan shift kerja karyawan berbasiskan *request* atau permintaan. Penelitian ini juga melakukan observasi terhadap kondisi aktual dalam pelaksanaan penjadwalan shift kerja di sebuah perusahaan. Data yang telah dikumpulkan kemudian dianalisis untuk mendapatkan permasalahan utama dan kebutuhan fungsional serta non fungsional sistem.

Implementasi algoritma SA dilakukan pada tahap pengembangan sistem, khususnya dalam modul *generate* jadwal kerja otomatis. Pada tahap ini, sistem akan menerima data karyawan, parameter jadwal, serta permintaan (*request*) *shift* yang telah disetujui *supervisor*, kemudian menggunakan algoritma SA untuk menghasilkan jadwal kerja yang optimal dengan meminimalkan pelanggaran terhadap *constraint* seperti: libur berturut-turut, transisi *shift* tidak ideal, atau jumlah libur yang melebihi batas. Parameter *initial temperature*, *final temperature*, *cooling rate*, dan jumlah iterasi digunakan sebagai kendali proses optimasi.

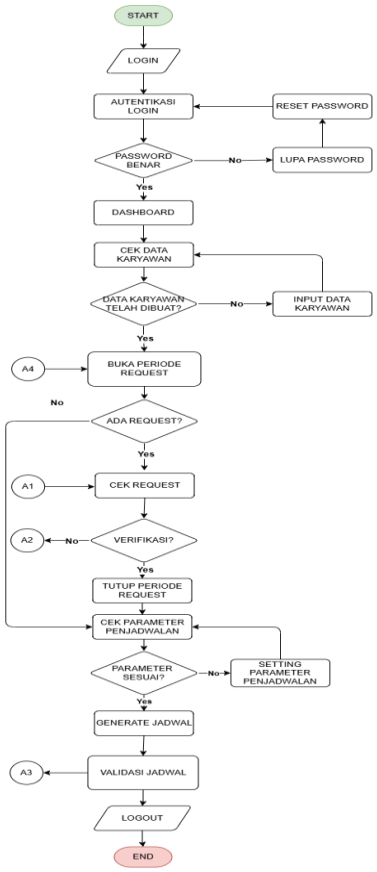
Penelitian ini menggunakan metode *waterfall* sebagai siklus hidup pengembangan perangkat lunak. Metode ini menggunakan pendekatan terstruktur yang mengharuskan setiap tahap diselesaikan secara berurutan sebelum melanjutkan ke tahap berikutnya [17] dan mampu menjadi acuan pengembangan perangkat lunak yang kebutuhannya tidak berubah selama periode pengembangan [18]. Metode ini terdiri dari lima tahapan utama, yaitu pendefinisian kebutuhan, desain sistem, implementasi serta pengujian unit, integrasi serta pengujian sistem, dan tahap operasi serta *maintenance* [19].

IV. HASIL PENELITIAN

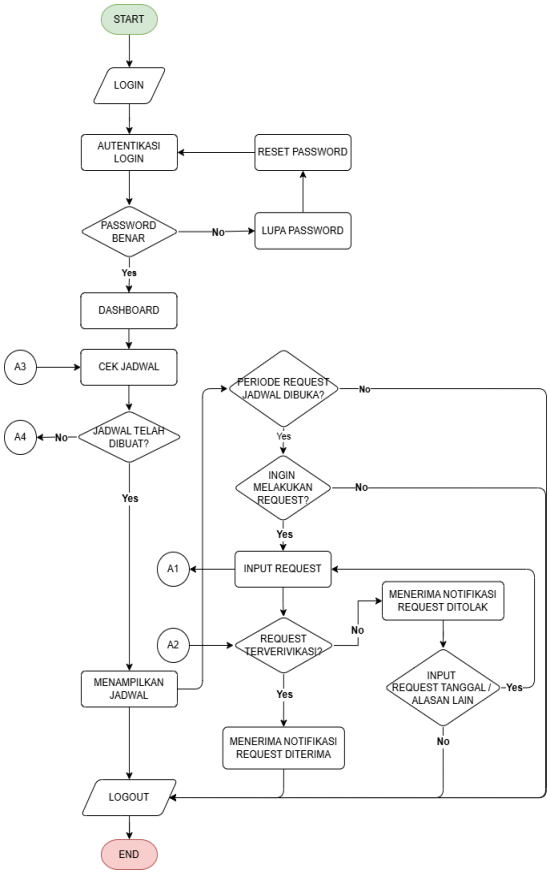
1. Kebutuhan Pengguna

Penelitian ini mengidentifikasi tiga aktor utama dalam penjadwalan shift kerja karyawan perusahaan, yaitu admin, *supervisor*, dan karyawan, dimana setiap aktor memiliki alur kerja masing-masing di dalam sistem. Gambar 1a menunjukkan aliran pengguna *supervisor*, yang memiliki fungsi utama seperti membuka periode *request* jadwal, memverifikasi *request* jadwal dari karyawan (*approve/reject*), *setting parameter*, serta melakukan *generate* jadwal berdasarkan parameter dan permintaan yang disetujui. Supervisor juga dapat memantau serta mengedit jadwal hasil generate jika dibutuhkan. Sementara itu, Gambar 1b menunjukkan aliran pengguna karyawan, yang berperan dalam mengajukan *request* jadwal, serta melihat status dari *request* jadwal yang diajukan dan melihat jadwal kerja final pada *dashboard*. Diagram *user flow* pada Gambar 1a dan Gambar 1b merupakan hasil tahapan kajian literatur dan observasi penjadwalan *shift* kerja berdasarkan *request* atau permintaan karyawan.

E-ISSN 2808-5841
P-ISSN 2808-7283



Gambar 1a.



Gambar 1b.

Gambar 1 *User Flow* diagram : (1a) m aktor supervisor (2b) aktor karyawan

Tabel 1.a Kebutuhan Fungsional

Kode	Kebutuhan Fungsional
KF-01	*Login*
KF-02	Mengelola Data Divisi
KF-03	Mengelola Data Jabatan
KF-04	Mengelola Data Supervisor
KF-05	Mengelola Data Karyawan
KF-06	Mengelola Jadwal
KF-07	*Logout*

Tabel 1.b Kebutuhan Non-Fungsional

Kode	Kebutuhan Non-Fungsional
KNF-01	Kebutuhan Keandalan / *Reliability*
KNF-02	Kebutuhan Ketersediaan / *Availability*
KNF-03	Kebutuhan Keamanan
KNF-04	Kebutuhan Pemeliharaan / *Maintainability*
KNF-05	Kebutuhan Kinerja / *Performance*
KNF-06	Atribut Kualitas Perangkat Lunak

Kebutuhan fungsional dan non-fungsional sistem telah dirangkum dalam Tabel 1.a dan Tabel 1.b. Pada proses penjadwalan shift, supervisor terlebih dahulu membuka periode *request*. Selanjutnya, karyawan mengajukan *request* jadwal sesuai preferensi. Setelah itu, supervisor memverifikasi dan menyetujui atau menolak *request* yang masuk. Setelah semua permintaan diverifikasi, *supervisor* dapat menjalankan proses *generate* jadwal menggunakan algoritma SA. Jadwal akhir yang dihasilkan kemudian dapat dilihat oleh karyawan.

Sistem penjadwalan *shift* kerja ini memiliki batasan dalam proses penjadwalannya, yaitu tidak diperbolehkan adanya pegawai yang libur dua hari berturut-turut, serta *shift* siang dan malam berurutan. Di sisi lain, terdapat tiga jadwal *shift* kerja yaitu *shift* pagi (08.00 – 16.00), siang (16.00 – 00.00), dan malam (00.00 – 08.00) Batasan diperoleh berdasarkan hasil observasi proses kerja di sebuah perusahaan yang menerapkan *shift* kerja.

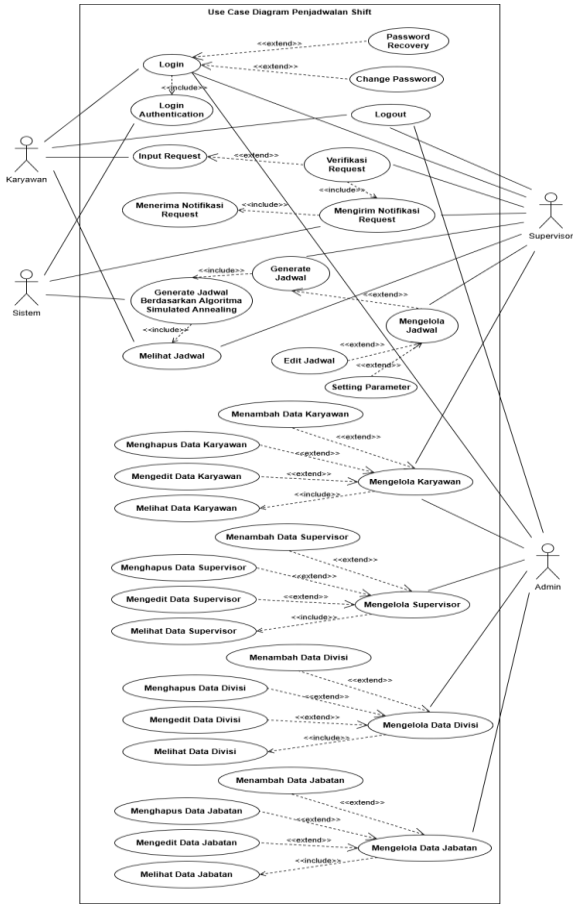
2. Desain Sistem

Berdasarkan kebutuhan fungsional dan nonfungsional, penelitian ini melakukan perancangan desain sistem menggunakan pemodelan *unified modelling language* (UML). Penelitian ini menghasilkan beberapa diagram, diantaranya adalah *use case diagram*, *activity diagram*, dan *sequence diagram*.

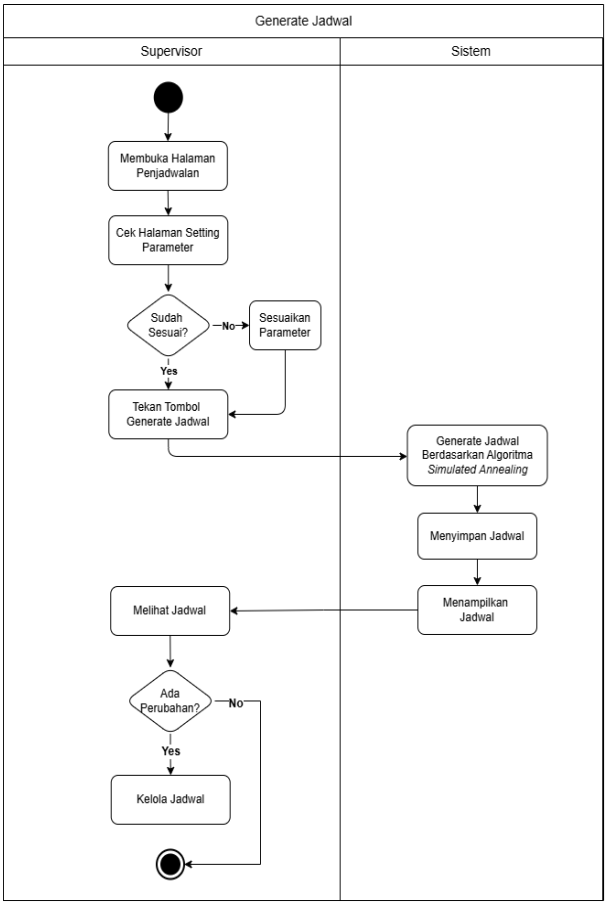
Diagram kasus penggunaan atau *use case diagram* pada penelitian ini ditunjukkan pada Gambar 2a yang memperlihatkan interaksi antara 4 aktor, yaitu *Admin*, *Supervisor*, Karyawan, dan Sistem. Terdapat 7 *base usecase* utama yang menggambarkan fungsionalitas inti dari sistem ini, yaitu: login, mengelola data divisi, jabatan, *supervisor*, karyawan, input request jadwal oleh karyawan, dan *geenerate* jadwal menggunakan algoritma *Simulated Annealing* oleh *supervisor*.

Berdasarkan 7 *base use case*, penelitian ini membuat 7 *activity diagram* yang mampu menunjukkan aktifitas aktor di dalam sistem untuk setiap skenario pada sebuah *use case*. Salah satu diagram aktifitas yang dihasilkan pada penelitian ini ditunjukkan pada Gambar 2b, yang menunjukkan aktifitas proses *generate* jadwal menggunakan algoritma SA berdasarkan permintaan atau *request* karyawan. Diagram ini menggambarkan alur aktivitas yang dilakukan oleh *Supervisor* dalam menjalankan fitur penjadwalan. Proses diawali dari pengecekan parameter, dilanjutkan dengan eksekusi *generate* jadwal, penyimpanan hasil ke *database*. jika diperlukan, *Supervisor* dapat melakukan pengelolaan ulang terhadap jadwal, seperti mengubah *shift* pada tanggal tertentu sesuai kebutuhan. Selanjutnya, *sequence diagram* yang dihasilkan pada penelitian ini berbasiskan pada *activity diagram*, dimana terdapat 7 *sequence diagram* dan salah satunya ditunjukkan pada Gambar 3.

E-ISSN 2808-5841
P-ISSN 2808-7283

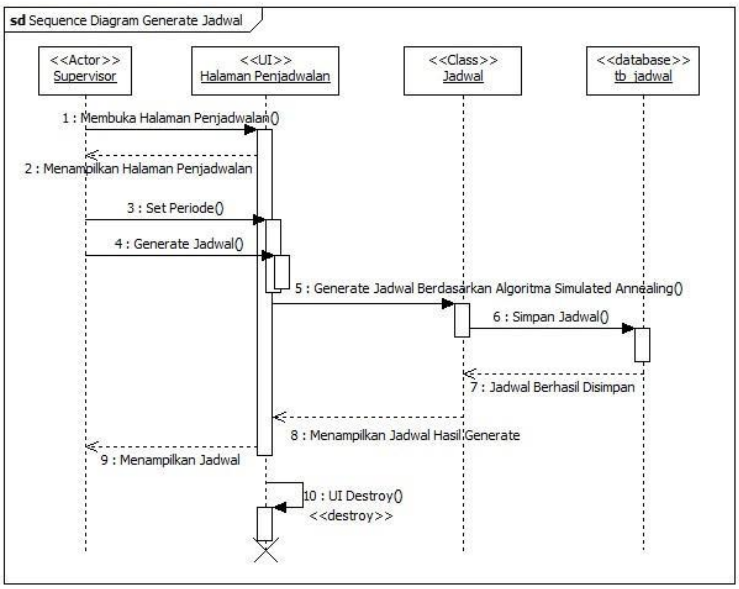


Gambar 2a. *Use Case* Diagram Penjadwalan *Shift*



Gambar 2b. *Activity* Diagram *Generate* Jadwal

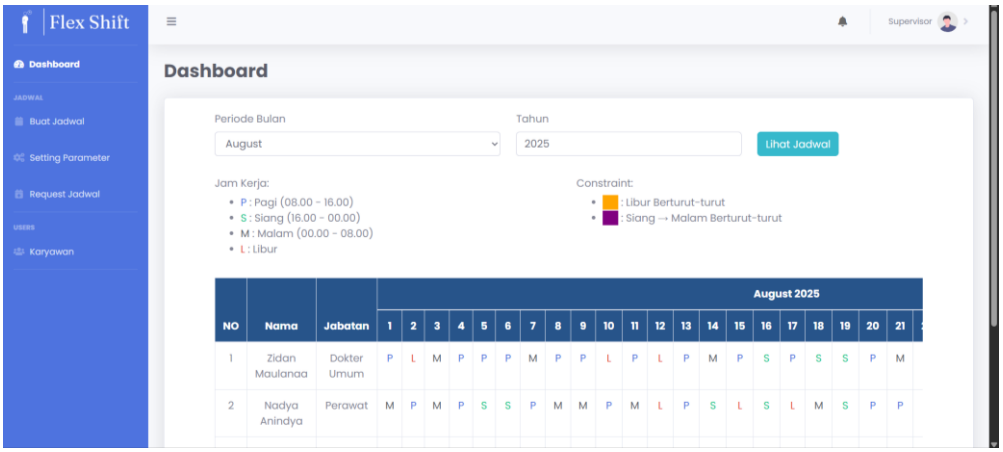
Gambar 2 Diagram Sistem : (2a) *use case* diagram (2b) *activity* diagram *generate* jadwal



Gambar 3 *Sequence* diagram *generate* jadwal

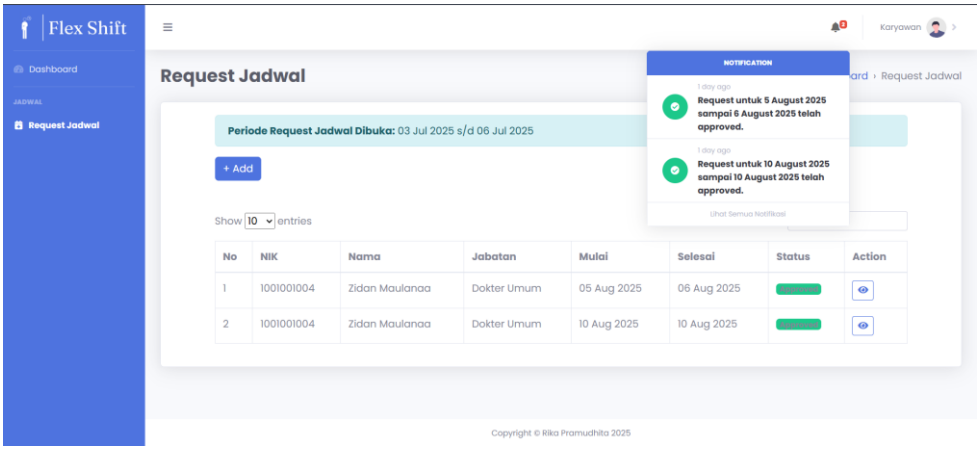
3. Implementasi Antarmuka Sistem

Pemodelan *system* yang digunakan untuk mengembangan sistem penjadwalan *shift* kerja karyawan berbasis *request* menggunakan algoritma SA. Pengembangan sistem berbasis *web* menggunakan *platform Laravel*. Setiap antarmuka yang dibangun disesuaikan dengan peran masing-masing pengguna yang telah didesain pada pemodelan UML sebelumnya. Gambar 4 menunjukkan implementasi antarmuka pembuatan jadwal oleh *supervisor*. *Supervisor* dapat melakukan *generate* jadwal dan hasil perhitungan algoritma SA ditunjukkan pada Gambar 5. Namun hasil *generate* ini bisa dilaksanakan apabila *request* atau permintaan karyawan telah diisi pada periode *request* yang telah ditentukan oleh supervisor. Proses permintaan jadwal oleh karyawan ditunjukkan pada Gambar 6.



Gambar 4 Antarmuka Pembuatan Jadwal

Gambar 5 Antarmuka *Generate* Jadwal Berdasarkan Algoritma SA



Gambar 6 Antarmuka *Request* Jadwal oleh Karyawan

4. Pengujian Sistem

Penelitian ini melakukan pengujian sistem dengan dua metode, yaitu *blackbox* dan *whitebox* testing. Pengujian *blackbox* merujuk pada pengujian perangkat lunak yang dilakukan berdasarkan spesifikasi fungsional tanpa memeriksa desain dan kode program, bertujuan untuk memastikan bahwa fungsi, masukan, dan keluaran perangkat lunak telah sesuai dengan spesifikasi yang ditetapkan [20]. Sedangkan, *whitebox testing* adalah pengujian perangkat lunak yang dilakukan dari segi desain dan kode program untuk memastikan bahwa fungsi masukan dan keluaran yang dihasilkan sesuai dengan spesifikasi yang dibutuhkan [20]. Pengujian sistem dalam penelitian ini dilakukan dengan dua pendekatan, yaitu *usability testing* menggunakan metode PSSUQ dan pengujian logika program menggunakan *path testing*.

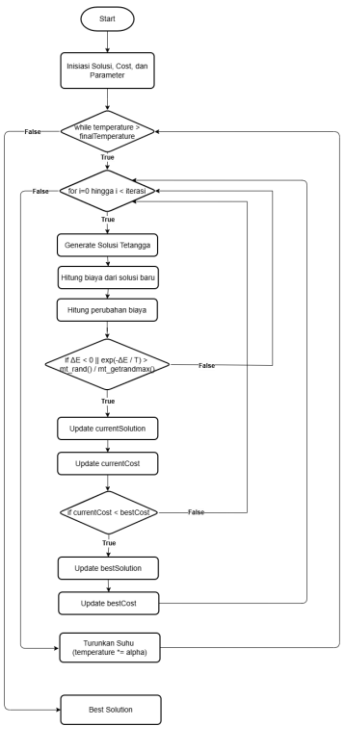
Pengujian *blackbox* dengan metode *usability* dilakukan setelah proses implementasi sistem selesai, dengan melibatkan 13 responden yang merupakan perwakilan pengguna. Metode yang digunakan adalah *Post-Study System Usability Questionnaire* (PSSUQ) dengan skala 1 hingga 7, di mana skor 1 menunjukkan kepuasan tertinggi dan skor 7 menunjukkan kepuasan terendah.

Tabel 2. Hasil Kuesioner PSSUQ

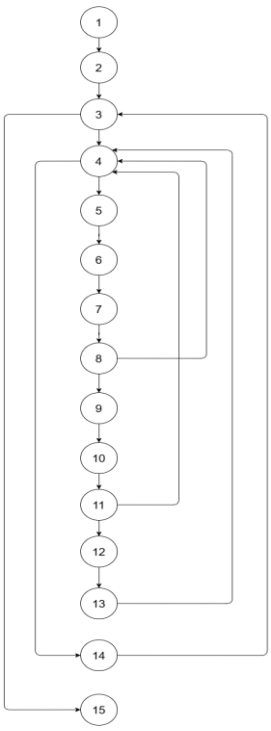
Index	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13
P1	1	1	1	2	1	2	2	2	2	2	1	1	1
P2	1	1	1	2	1	2	1	2	3	2	1	1	1
P3	1	2	1	1	1	1	2	2	2	3	2	1	1
P4	1	2	1	2	1	2	2	2	4	2	1	1	1
P5	2	2	2	2	2	2	2	2	2	2	1	1	2
P6	1	2	1	2	1	2	2	2	3	3	1	2	2
P7	1	2	1	2	1	1	2	2	2	2	1	1	1
P8	1	2	1	2	1	2	2	2	3	2	1	2	1
P9	1	2	1	2	1	1	2	2	2	2	1	1	1
P10	2	1	1	2	1	1	1	2	2	3	2	1	1
P11	2	2	1	2	1	2	2	2	3	2	2	2	2
P12	1	2	1	2	1	1	1	2	3	2	1	1	1
P13	1	1	2	2	1	1	1	2	3	2	1	1	1
P14	1	2	2	2	1	1	2	2	3	2	1	1	1
P15	1	2	2	2	1	2	2	2	3	5	1	1	1
P16	1	1	1	2	1	2	2	2	2	2	1	1	1

Hasil dari kuesioner PSSUQ pada Tabel 2, menunjukkan bahwa sistem memiliki tingkat *usability* yang sangat baik. Komponen *System Usefulness* (SYSUSE) memperoleh nilai dengan rata-rata 1,64, mengindikasikan bahwa sistem dinilai berguna dan berfungsi sebagaimana mestinya. Komponen *Information Quality* (INFOQUAL) memperoleh nilai dengan rata-rata 1,59, yang menunjukkan informasi dalam sistem mudah dipahami. Komponen *Interface Quality* (INTQUAL) memperoleh nilai dengan rata-rata 1,67, yang menunjukkan tampilan sistem cukup memuaskan. Komponen OVERALL dari seluruh 16 pertanyaan PSSUQ menghasilkan nilai dengan rata-rata 1,61, yang berarti sistem diterima dengan sangat baik oleh pengguna dari aspek fungsi, tampilan, dan kemudahan penggunaan.

Pengujian *white-box* dilakukan menggunakan metode *Path Testing* untuk mengevaluasi logika program pada fitur utama sistem, yaitu proses *generate()* dalam penjadwalan *shift* kerja dengan algoritma *Simulated Annealing*. Fokus pengujian adalah memastikan bahwa seluruh cabang logika dalam fungsi tersebut berjalan sesuai dengan spesifikasi yang ditentukan.



Gambar 7a. *Flowchart Function Generate*



Gambar 7b. *Flowgraph Function Generate*

Gambar 7a dan Gambar 7b masing-masing menunjukkan *flowchart dan flowgraph* dari fungsi *generate()*. Dari Gambar 7a, diketahui *node* (N) berjumlah 15, *edge (E)* berjumlah 18, dan *predicate node* (P) berjumlah 4. Setelah diketahui nilai dari *flowgraph*, maka untuk perhitungan *cyclomatic complexity V(G)* adalah :

- V(G) = E – N + 2 = 18 – 15 + 2 = 5
- V(G) = P + 1 = 4 + 1 = 5

Berdasarkan hasil tersebut, didapatkan jalur indepen sebagai berikut :

1. 1 - 2 - 3(*true*) - 4(*true*) - 5 - 6 - 7 -8(*true*) - 9 - 10 - 11(*true*) - 12 - 13 - 4(*next iteration*)
2. 1 - 2 - 3(*true*) - 4(*true*) - 5 - 6 - 7 - 8(*true*) - 9 - 10 - 11(*false*) - 4 (*next iteration*)
3. 1 - 2 - 3(*true*) - 4(*true*) - 5 - 6 - 7 - 8(*false*) - 4 (*next iteration*)
4. 1 - 2 - 3(*true*) - 4(*false*) - 14 - 3 (*next while loop*)
5. 1 - 2 - 3(*false*) - 15 (*stop while loop*)

Tabel 3. Test Case

Jalur	Kegiatan	Hasil yang diharapkan	Hasil	Keterangan
1	Menerima solusi baru karena lebih baik	Solusi baru diterima karena lebih baik (semua kondisi bernilai true), lalu *update best solution & best cost*	Sesuai	Valid
2	Jika *delta(E)* lebih kecil daripada 0 atau diterima via *probabilitas* Jika *cost* baru dari sebuah solusi nilainya lebih besar dari *best cost.*	Solusi baru diterima namun *best solution* dan *best cost* tidak di *update* (bernilai *false*)	Sesuai	Valid
3	Jika *delta* lebih besar atau sama dengan 0 dan tidak memenuhi probabilitas.	Solusi ditolak dan *current solution* dan *current cost* tetap	Sesuai	Valid
4	Jika iterasi untuk satu suhu telah selesai, suhu diturunkan kemudian melanjutkan iterasi untuk suhu yang baru.	Iterasi selesai dan *update* suhu baru	Sesuai	Valid
5	Jika suhu telah mencapai final *temperature*	Solusi terbaik telah didapatkan	Sesuai	Valid

Pengujian dilakukan dengan menelusuri setiap jalur eksekusi logika program berdasarkan *flowgraph.* Hasil pengujian menunjukkan bahwa semua kondisi telah berhasil dieksekusi sesuai dengan spesifikasi. Dengan demikian, logika algoritma *Simulated Annealing* dalam fungsi *generate()* telah berjalan valid dan sesuai harapan.

V. KESIMPULAN

Penelitian ini berhasil mengembangkan sistem penjadwalan *shift* kerja karyawan berdasarkan permintaan jadwal oleh karyawan dengan menggunakan algoritma *Simulated Annealing* (SA). Sistem yang dikembangkan juga dilengkapi fitur fleksibilitas yang memungkinkan karyawan mengajukan permintaan *shift,* dan libur, serta memungkinkan *supervisor* untuk menyetujui atau menolak permintaan tersebut.

Hasil implementasi menunjukkan bahwa sistem mampu menghasilkan jadwal yang mempertimbangkan *constraint* utama, seperti larangan libur berturut-turut dan transisi *shift* siang ke malam. Dari total 4 permintaan *shift* yang diajukan oleh karyawan, 100% *request* yang disetujui berhasil dimasukkan ke dalam hasil jadwal akhir. Pengujian *whitebox* dengan metode *path testing* menghasilkan 5 jalur independen yang menunjukkan logika algoritma berjalan sesuai spesifikasi. Sementara itu, pengujian *usability* menggunakan kuesioner PSSUQ terhadap 13 responden menghasilkan skor rata-rata 1,61 (dari skala 1–7), yang mengindikasikan tingkat kepuasan pengguna sangat tinggi terhadap sistem dari aspek fungsi, tampilan, dan kemudahan penggunaan.

Penelitian ini memberikan kontribusi dalam penerapan algoritma SA pada studi kasus penjadwalan kerja dengan permintaan karyawan, dan dapat menjadi referensi bagi instansi atau perusahaan yang ingin menerapkan penjadwalan kerja fleksibel berbasis permintaan karyawan secara lebih optimal. Namun, pada beberapa kasus awal saat proses *generate,* jadwal pertama yang dihasilkan masih mengandung pelanggaran *constraint.* Penelitian selanjutnya disarankan untuk mengeksplorasi pengaturan parameter algoritma untuk meningkatkan stabilitas dan kualitas hasil penjadwalan, serta mengembangkan dukungan terhadap jenis *shift* yang lebih kompleks dan integrasi dengan sistem kehadiran untuk penerapan yang lebih luas.

REFERENSI

[1] T. Bustami and S. Alam, 'Penerapan Teknologi Informasi dalam Manajemen Operasional Analisis Implementasi di PT. Unilever Tbk', *SEIKO : Journal of Management and Business*, vol. 7, no. 1, pp. 1321–1329, 2024.

[2] S. A. Darmawan, I. Cholissodin, and Tibyani, 'Optimasi Penjadwalan Mesin dan Shift Karyawan Menggunakan Algoritme Genetika (Studi Kasus Pada PT. Petro Jordan Abadi)', *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*, vol. 2, no. 12, pp. 6793–6801, 2018.

[3] W. Priatna, J. Warta, and D. Sulistiyo, 'Implementasi Algoritma Genetika untuk Aplikasi Penjadwalan Sistem Kerja Shift', *tc*, vol. 22, no. 1, pp. 235–246, Feb. 2023, doi: 10.33633/tc.v22i1.7049.

[4] J. Mustofa, 'Sistem Informasi Penjadwalan Shift Kerja Karyawan Menggunakan Algoritma Genetika (Studi Kasus Jawara Digital Art Store Yogjakarta)', Universitas Teknologi Yogyakarta, Yogyakarta, 2020.

[5] L. W. Santoso, J. Guntara, and I. N. Sandjaja, "PENJADWALAN PRODUKSI DENGAN MENGGUNAKAN ALGORITMA SIMULATED ANNEALING," Jurnal Ilmiah Ilmu Komputer, vol. 9, no. 1, pp. 1–6, Sep. 2012.

[6] H. A. Shiddiq, Sugiono, and C. F. M. Tantrika, "IMPLEMENTASI ALGORITMA SIMULATED ANNEALING PADA PENJADWALAN PRODUKSI UNTUK MEMINIMASI MAKESPAN (Studi Kasus di PT. Gatra Mapan, Karang Ploso, Malang)," Jurnal Rekayasa dan Manajemen Sistem Industri, vol. 3, no. 1, 2015.

[7] D. S. Rahayu, A. Suryapratama, A. Z. Amongsaufa, and B. I. K. Koloay, "EVALUASI ALGORITMA RUNUT BALIK DAN SIMULATED ANNEALING PADA PERMAINAN SUDOKU," JuTISI, vol. 3, no. 1, Apr. 2017, doi: 10.28932/jutisi.v3i1.592.

[8] E. N. Ananti, I. Cholissodin, and B. Rahayudi, "Optimasi Penjadwalan Pekerja Shift di Rumah Makan Cepat Saji (Fast Food Restaurant) menggunakan Algoritma Genetika (Studi Kasus: Warung Gunung di Kediri)," Jul. 2021.

[9] A. Josi, "Implementasi Algoritma Genetika pada Aplikasi Penjadwalan Perkuliahan Berbasis Web dengan mengadopsi model Waterfall (Studi Kasus: STMIK Prabumulih)," JPIT, vol. 2, no. 2, pp. 77–83, Jul. 2017, doi: 10.30591/jpit.v2i2.517.

[10] M. N. Cahya, I. E. Maulani, I. Intan, and T. A. Ambarwati, "Penerapan Algoritma Genetika dalam Optimisasi Penjadwalan Sistem Informasi Akademik," Jurnal Sosial Teknologi, vol. 3, no. 2, pp. 103–107, 2023.

[11] R. Fikria and R. P. Sari, 'Sistem Penjadwalan Piket Pegawai di Dinas Pencegahan dan Penyelamatan Kota Medan', *INNOVATIVE: Journal of Social Science Research*, vol. 3, no. 6, pp. 5744–5757.

[12] Suseno and E. Shuha, 'Penjadwalan Tenaga Kerja untuk Tiga Shift Kerja dengan Pengembangan Metode Algoritma Tibrewala, Philippe, dan Browne', presented at the Seminar Nasional Teknik Industri (SNTI2017), 2017, pp. 298–300.

[13] V. Marchelia, 'Stres Kerja ditinjau dari Shift Kerja pda Karyawan', *JIPT*, vol. 2, no. 2, p. 1, 2014.

[14] E. Febianti, A. I. Saeful M, and J. Fitra, 'Usulan Penjadwalan Produksi Baja Profil Menggunakan Metode Nawaz Enscore And Ham dan Algoritma Simulated Annealing', presented at the Seminar Nasional Sains dan Teknologi 2019, Jakarta: Univeritas Muhammadiyah Jakarta, 2019, pp. 1–9.
[15] R. Hidayati, I. Guntoro, and S. Junianti, 'Penggunaan Metode Simulated Annealing untuk Penyelesaian Travelling Salesman Problem', *Journal of Computer Engineering System and Science*, vol. 4, no. 2, pp. 217–221, 2019.
[16] H. A. Prawira and B. Santosa, 'Pengembangan Algoritma Particle Swarm Optimization dan Simulated Annealing untuk Menyelesaikan Vehicle Routing Problem with Drone', *prozima*, vol. 5, no. 1, pp. 1–11, Jul. 2021, doi: 10.21070/prozima.v5i1.1398.
[17] W. Nugraha, M. Syarif, and W. S. Dharmawan, 'Penerapan Metode SDLC Waterfall dalam Sistem Informasi Inventory Barang berbasis Desktop', *jusim*, vol. 3, no. 1, pp. 22–28, Jun. 2018, doi: 10.32767/jusim.v3i1.246.
[18] I. Dzikria and A. Rizal, 'Rancang Bangun Sistem Pemesanan Mandiri Restoran Berbasis Progressive Web Apps', *Jurnal Sistim Informasi dan Teknologi*, vol. 5, no. 1, pp. 135–144, 2023.
[19] Y. Wahyudin and D. N. Rahayu, 'Analisis Metode Pengembangan Sistem Informasi Berbasis Website: A Literatur Review', *interkom*, vol. 15, no. 3, pp. 26–40, Oct. 2020, doi: 10.35969/interkom.v15i3.74.
[20] W. N. Cholifah, Yulianingsih, and S. M. Sagita, 'Pengujian black box testing pada aplikasi action & strategy berbasis android dengan teknologi phonegap', *STRING (Satuan Tulisan Riset dan Inovasi Teknologi)*, vol. 3, no. 2, pp. 206–210, 2018.