IMPLEMENTASI ALGORITMA *SELECTION SORT* DALAM SISTEM ABSENSI SISWA UNTUK PENGURUTAN KEAKTIFAN BERDASARKAN KEHADIRAN (STUDI KASUS : SMA NEGERI 1 NAWANGAN PACITAN)

Herdy Pramudya, Adi Fajaryanto Cobantoro, Jamilah Karaman
Teknik Informatika, Universitas Muhammadiyah Ponorogo
Jl. Budi Utomo No.10, Ponorogo, Jawa Timur, Indonesia
herdypramuedya12@gmail.com

ABSTRAK

Absensi siswa merupakan aspek penting dalam administrasi sekolah yang berpengaruh terhadap evaluasi akademik dan nonakademik. Pengelolaan absensi secara manual sering menimbulkan kendala, seperti kesalahan pencatatan dan lambatnya analisis data, sehingga memengaruhi akurasi serta ketepatan pengambilan keputusan. Penelitian ini bertujuan mengimplementasikan algoritma *Selection Sort* dalam sistem absensi siswa di SMA Negeri 1 Nawangan, yang masih mengalami kesulitan dalam pengelolaan data absensi secara manual karena memerlukan waktu lama dan rentan terhadap kesalahan. Solusi yang ditawarkan adalah sistem absensi digital yang mampu mengurangi kesalahan dan mempercepat pengolahan data. Penelitian ini menggunakan metode pengembangan *waterfall* yang mencakup tahap perencanaan, analisis, desain, implementasi, dan pengujian. Hasil penelitian menunjukkan bahwa algoritma *Selection Sort* dapat diintegrasikan secara optimal untuk mengurutkan data absensi siswa berdasarkan kriteria tertentu, seperti nama, kelas, atau tingkat keaktifan. Pengujian *whitebox* menunjukkan bahwa algoritma ini memiliki akurasi tinggi, dengan perhitungan *Cyclomatic Complexity* sebesar 3, yang menandakan struktur kontrol sederhana dan mudah dipahami. Implementasi ini menghasilkan sistem absensi yang lebih terorganisir dan andal.

Kata kunci : *Algoritma Selection Sort, Absensi Siswa, Keaktifan Siswa, Sistem Absensi.*

1. PENDAHULUAN

Absensi merupakan pendataan kehadiran seseorang dalam suatu aktivitas di sebuah institusi, di mana absensi siswa menjadi bagian penting dari administrasi sekolah karena berdampak langsung pada evaluasi akademik dan non-akademik[1][2].

Kehadiran siswa yang konsisten sering kali digunakan sebagai indikator keaktifan dan komitmen siswa terhadap kegiatan belajar mengajar di sekolah[3].

Hal ini didukung oleh [4] menyatakan bahwa data absensi memainkan peran kunci dalam memantau perkembangan siswa, membantu guru dan staf sekolah dalam mengidentifikasi siswa yang membutuhkan perhatian lebih terkait kehadiran siswa. Dengan berkembangnya teknologi informasi, semakin banyak sekolah yang beralih ke sistem absensi digital untuk meningkatkan akurasi, efisiensi, dan kemudahan akses terhadap data siswa secara *real-time*[5].

Berdasarkan studi pendahuluan yang dilakukan melalui wawancara dengan guru dan staf administrasi di SMA Negeri 1 Nawangan, terungkap bahwa siswa sering menghadapi masalah dalam pengelolaan data kehadiran siswa. Staf administrasi menyebutkan bahwa proses penyusunan daftar keaktifan siswa memakan waktu lama, terutama menjelang akhir semester ketika data perlu diolah untuk keperluan evaluasi. Selain itu, sering terjadi ketidaksesuaian dalam pencatatan kehadiran akibat kesalahan manual dalam proses pengelolaan data absensi. Implementasi sistem berbasis algoritma seperti Selection Sort diharapkan dapat meminimalkan kesalahan ini dan mempercepat proses pengurutan data kehadiran siswa.

SMA Negeri 1 Nawangan menghadapi kendala dalam pengelolaan data kehadiran siswa yang masih dilakukan secara manual. Proses pencatatan menggunakan lembar kertas mengharuskan guru mencatat kehadiran secara tertulis, kemudian data direkapitulasi oleh pihak sekolah. Pendekatan ini memakan waktu dan rentan terhadap kesalahan, seperti data yang hilang atau tidak akurat. Kesulitan lain adalah lambatnya identifikasi pola keaktifan siswa karena analisis data harus dilakukan secara manual, terutama dengan jumlah siswa yang besar. Implementasi teknologi yang lebih optimal diperlukan untuk meningkatkan efisiensi dan akurasi pengelolaan data tersebut.

Sistem manual ini juga menghambat efisiensi dalam penilaian kehadiran siswa, sehingga mempersulit pengambilan keputusan guna penambahan nilai para siswa terutama dalam menindaklanjuti keaktifan siswa secara cepat dan tepat. Selain nilai UTS dan UAS, nilai keaktifan siswa dalam kelas dapat menjadi indikator penting untuk evaluasi secara menyeluruh. Dengan demikian, penilaian yang mencakup aspek keaktifan dapat memberikan gambaran lebih mendalam mengenai keterlibatan dan kedisiplinan siswa. Hasil penilaian ini berpotensi menjadi dasar dalam pengambilan keputusan untuk memberikan nilai tambahan sebagai bentuk apresiasi terhadap siswa yang menunjukkan keaktifan dan kedisiplinan. Sementara itu, bagi siswa yang memiliki tingkat keaktifan rendah, dapat

diberikan teguran oleh guru Bimbingan dan Konseling (BK) sebagai upaya pembinaan. Sistem absensi digital telah terbukti mampu mengurangi kesalahan pencatatan dan mempercepat proses pengumpulan serta pengolahan data[6].

Untuk mengatasi masalah tersebut, SMA Negeri 1 Nawangan dengan menggunakan sistem absensi berbasis web yang didukung oleh algoritma Selection Sort untuk mengurutkan data kehadiran siswa. Algoritma ini bekerja dengan menyusun kehadiran siswa berdasarkan tingkat keaktifan, sehingga pihak sekolah dapat dengan mudah mengidentifikasi siswa yang paling sering hadir maupun yang memiliki tingkat kehadiran rendah. Sistem ini juga memberikan kemudahan dalam menghasilkan laporan kehadiran secara real-time, mengurangi kesalahan pencatatan manual, serta mempercepat proses pengambilan keputusan terkait kedisiplinan siswa. Dengan adanya sistem ini, pengelolaan absensi menjadi lebih teratur dan akurat.

2. TINJAUAN PUSTAKA

Setelah Dalam rangka memahami dan menganalisis pengurutan keaktifan siswa berdasarkan kehadiran di SMA Negeri 1 Nawangan Pacitan menggunakan algoritma *Selection Sort*, tinjauan pustaka ini akan mengkaji berbagai konsep dan teori yang mendukung penelitian ini. Tinjauan pustaka ini bertujuan untuk memberikan dasar teoritis yang kuat bagi penelitian yang dilakukan, serta menunjukkan bagaimana penelitian ini berkontribusi dalam mengatasi permasalahan dalam sistem absensi siswa. Dengan demikian, diharapkan landasan teori yang kuat ini dapat mendukung penelitian dalam implementasi algoritma *Selection Sort* untuk pengurutan keaktifan berdasarkan kehadiran, sehingga sistem absensi siswa dapat dikembangkan secara lebih terarah dan mendalam.

2.1. Penelitian Terdahulu

Setelah mengkaji beberapa penelitian, ditemukan bahwa beberapa di antaranya relevan dengan penelitian yang sedang dilakukan. Penelitian oleh menerapkan algoritma *Selection Sort* untuk pengurutan surat keluar pada website Sistem Informasi Desa Tugu Mlarak menggunakan PHP dan *Framework* Laravel, yang meningkatkan akurasi serta kemudahan pencarian dan pengelolaan surat.

Selanjutnya, penelitian oleh [7] mengimplementasikan *Selection Sort* untuk pengurutan nilai IPK mahasiswa Universitas Potensi Utama menggunakan PHP, yang terbukti mampu mengurutkan nilai dengan cepat dan akurat, membantu administrasi akademik dalam pengelolaan data.

Kemudian, penelitian oleh [8] mengaplikasikan algoritma yang sama pada sistem informasi retensi dokumen rekam medis di Klinik PKU Muhammadiyah, yang berfungsi untuk mengurutkan dokumen berdasarkan kunjungan terakhir guna memudahkan proses retensi.

Selain itu, penelitian oleh [9] menggunakan *Selection Sort* untuk perangkingan poin pada *E-Sports Tournament* Garuda *League* dengan PHP, di mana algoritma ini berhasil mengurutkan posisi tim secara *descending* di website resmi liga.

Terakhir, penelitian oleh [10] membandingkan *Selection Sort* dan *Insertion Sort* dalam pengurutan data menggunakan bahasa pemrograman C dan Java, yang menunjukkan bahwa *Selection Sort* lebih sederhana dan cepat karena tidak memerlukan variabel penampung atau proses geser data. Dari kajian ini, dapat disimpulkan bahwa algoritma *Selection Sort* memiliki berbagai penerapan dalam sistem informasi dan mampu meningkatkan efisiensi dalam proses pengurutan data.

2.2. Kehadiran / Absensi

Kehadiran atau absensi merupakan salah satu indikator penting dalam mengevaluasi keterlibatan individu dalam suatu kegiatan, baik di lingkungan akademik maupun di tempat kerja. Kehadiran yang baik sering kali dianggap sebagai cerminan dari komitmen, tanggung jawab, dan partisipasi aktif seseorang terhadap tugas atau kewajiban yang diembannya[11].

Salah satu indikator kemajuan suatu negara adalah kualitas pendidikannya, yang tercermin dalam tingkah laku, sikap, dan sifat masyarakatnya[12].

Kehadiran siswa di kelas berperan penting dalam pencapaian akademik dan keterlibatan belajar, di mana penelitian menunjukkan bahwa siswa dengan tingkat kehadiran tinggi cenderung meraih prestasi lebih baik dibandingkan yang sering absen[13].

Penelitian menunjukkan bahwa siswa yang memiliki tingkat kehadiran tinggi cenderung memiliki prestasi yang lebih baik dibandingkan siswa yang sering absen. Ini karena kehadiran yang konsisten memberikan kesempatan lebih besar untuk terlibat dalam aktivitas pembelajaran, interaksi dengan pengajar, serta akses langsung terhadap materi dan diskusi yang terjadi di kelas[4].

2.3. *Website*

*Web*site secara umum adalah kumpulan halaman *web* yang saling terhubung dan dapat diakses melalui internet menggunakan sebuah browser. Halaman-halaman ini biasanya berisi informasi berupa teks, gambar, video, atau konten multimedia lainnya. *Web*site disusun dengan tujuan tertentu seperti memberikan informasi, menyediakan layanan, atau menjual produk. Setiap *web*site memiliki alamat unik yang dikenal sebagai URL (*Uniform Resource Locator*), yang memudahkan pengguna untuk mengaksesnya[14].

2.4. Algoritma *Selection Sort*

Selection Sort adalah algoritma yang bersifat in-place, artinya pengurutan dilakukan langsung pada *array* tanpa membutuhkan struktur data tambahan dengan memori yang signifikan. Ini menjadikannya

efisien dalam penggunaan ruang memori. Namun, meskipun sederhana dan mudah dipahami, algoritma ini kurang optimal dalam hal kecepatan untuk dataset yang besar karena kompleksitas waktu yang tinggi yaitu $O(n^2)$. *Selection Sort* tidak tergantung pada apakah data awal sudah sebagian terurut atau tidak; ia tetap melakukan jumlah perbandingan yang sama dalam semua kasus[15].

Selection Sort juga memiliki sifat *unstable*, artinya ketika ada elemen-elemen yang bernilai sama, urutan relatif antara elemen-elemen tersebut bisa berubah setelah pengurutan. Meskipun begitu, algoritma ini berguna dalam situasi di mana penggunaan memori menjadi pertimbangan utama, atau ketika ukuran dataset relatif kecil sehingga performa $O(n^2)$ masih dapat diterima. Selain itu, karena pola kerjanya yang sangat sistematis, algoritma ini mudah diimplementasikan dan di-debug[10].

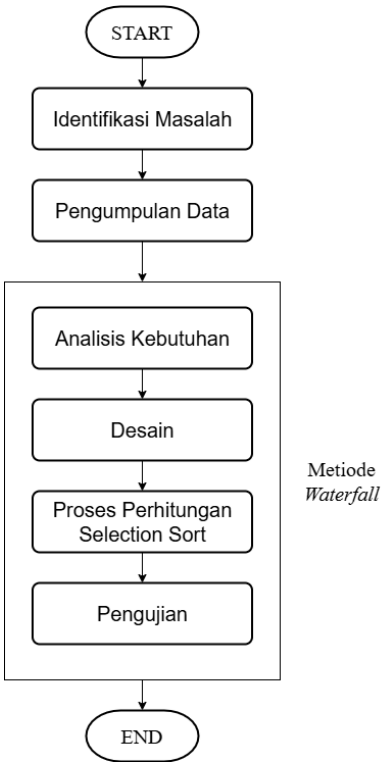
2.5. *Whitebox Testing*

Whitebox testing adalah metode pengujian perangkat lunak yang mengevaluasi struktur internal atau kode sumber dari aplikasi. Dalam pendekatan ini, penguji memiliki akses ke logika program dan meneliti cara kerja sistem di tingkat kode. Tujuan utama *white box testing* adalah memastikan bahwa alur logika, cabang, *loop*, dan setiap jalur dalam kode berfungsi sesuai dengan spesifikasi yang telah ditetapkan dan bebas dari kesalahan[16].

White box testing mencakup beberapa teknik, seperti *statement coverage*, *branch coverage*, *path coverage*, dan *loop testing*. *Statement coverage* memastikan setiap pernyataan dalam kode dieksekusi setidaknya sekali. *Branch coverage* berfokus pada pengujian setiap cabang atau kondisi dalam kode untuk memastikan bahwa semua kemungkinan jalur logika diuji. *Path coverage* adalah pengujian yang lebih komprehensif, di mana setiap jalur yang mungkin dilalui dalam program dievaluasi. *Loop testing* menargetkan berbagai kondisi dalam perulangan untuk memastikan bahwa batas dan logika perulangan berfungsi dengan benar, terutama pada nilai batas yang ekstrem[17].

3. METODE PENELITIAN

Berdasarkan tahapan penelitian pada gambar 1, *flowchart* tersebut menunjukkan alur proses yang menggunakan pendekatan metode *Waterfall*, sebuah model pengembangan perangkat lunak yang sistematis dan berurutan, di mana setiap tahap harus diselesaikan sebelum melanjutkan ke tahap berikutnya[18]. Tahapan tersebut dimulai dengan Identifikasi Masalah, yang bertujuan untuk memahami permasalahan utama, dilanjutkan dengan Pengumpulan Data untuk mendukung analisis, kemudian Analisis Kebutuhan untuk memastikan solusi yang dikembangkan sesuai dengan tujuan. Selanjutnya, pada tahap Desain, rancangan sistem dibuat mencakup arsitektur dan antarmuka, diikuti dengan tahap Implementasi yang menerapkan desain ke dalam bentuk sistem yang dapat dijalankan, dan akhirnya pada tahap Pengujian, dilakukan verifikasi dan validasi untuk memastikan sistem berfungsi sesuai spesifikasi.



Gambar 1. Tahapan Metodologi

Pertama kali metode *waterfall* diperkenalkan oleh Herbert D. Benington beliau mempresentasikan konsep yang mirip dengan model *Waterfall* pada tahun 1956 dalam presentasinya di *Symposium on Advanced Programming Method for Digital Computers*, dengan fokus pada pengembangan perangkat lunak untuk SAGE (*Semi Automatic Ground Environment*). Namun, Winston W. Royce yang lebih dikenal secara luas karena secara formal memperkenalkan model *Waterfall* dalam bukunya "*Managing the Development of Large Software Systems*" pada tahun 1970[19]. Royce menggambarkan pendekatan linier yang terdiri dari tahap-tahap yang jelas dan berurutan dalam pengembangan perangkat lunak, yang kemudian dikenal sebagai model *Waterfall* dan menjadi pendekatan modern pertama yang banyak digunakan dalam pembangunan sistem perangkat lunak.

3.1. Identifikasi Masalah

Tahap identifikasi masalah dilakukan melalui observasi, wawancara, dan analisis sistem absensi di SMA Negeri 1 Nawangan Pacitan untuk memahami kendala yang dihadapi. Masalah utama yang ditemukan adalah pengelolaan data kehadiran yang tidak terstruktur serta kesulitan dalam mengurutkan siswa berdasarkan tingkat keaktifan. Untuk mengatasi hal ini, dilakukan studi literatur mengenai algoritma *Selection Sort* guna memastikan kecocokannya dalam proses pengurutan data kehadiran siswa. Data yang dikumpulkan kemudian digunakan sebagai contoh

kasus dalam implementasi sistem absensi berbasis algoritma tersebut.

3.2. Pengumpulan Data

Tahap pengumpulan data dilakukan melalui studi literatur serta observasi dan wawancara di SMA Negeri 1 Nawangan Pacitan. Studi literatur bertujuan memahami penerapan *Selection Sort* dalam sistem absensi siswa melalui analisis buku, jurnal, dan penelitian terkait, dengan fokus pada pengurutan keaktifan siswa berdasarkan kehadiran. Sementara itu, observasi dan wawancara dilakukan untuk mengkaji proses absensi yang berjalan, kendala dalam pengelolaan data, serta kebutuhan sistem yang lebih otomatis. Hasil dari kedua metode ini menjadi dasar dalam merancang sistem absensi berbasis *Selection Sort* untuk mempermudah pengurutan data siswa berdasarkan kehadiran.

3.3. Analisis Kebutuhan

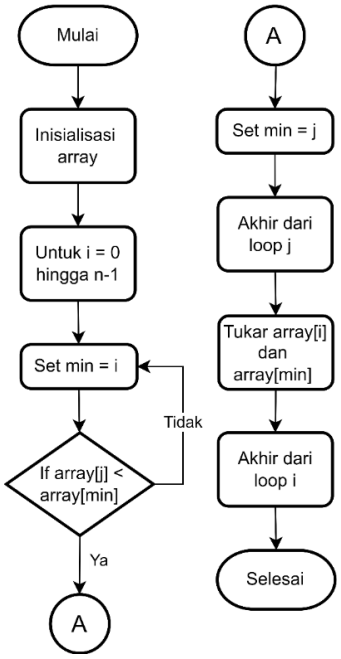
Dalam metodologi *Waterfall*, sistem absensi siswa di SMA Negeri 1 Nawangan Pacitan dirancang dengan tiga jenis pengguna utama admin, guru, dan siswa. Kebutuhan fungsional mencakup fitur seperti registrasi, *login*, pengelolaan data kehadiran dan tugas, serta pemantauan hasil pengurutan menggunakan algoritma *Selection Sort*. Admin bertugas mengelola data siswa dan absensi, guru memantau kehadiran serta memberikan nilai, sementara siswa melakukan absensi dan mengunggah tugas. Kebutuhan non-fungsional meliputi sistem berbasis web yang dibangun dengan PHP, Laravel, dan MySQL, antarmuka yang ramah pengguna, serta dukungan akses dari berbagai perangkat. Dengan pendekatan *Waterfall*, setiap tahapan perancangan dilakukan secara terstruktur untuk memastikan sistem berjalan optimal dan mampu mengelola data kehadiran siswa dengan lebih sistematis.

3.4. Desain

Tahap desain dalam metodologi *Waterfall* untuk sistem absensi siswa di SMA Negeri 1 Nawangan Pacitan mencakup *Flowchart Selection Sort*, diagram konteks, *usecase diagram* serta ERD dan relasi tabel untuk pengelolaan data siswa dan kehadiran. Desain ini bertujuan memberikan kerangka kerja yang jelas agar implementasi berjalan sesuai rencana, memungkinkan sistem mengurutkan keaktifan siswa secara optimal berdasarkan tingkat kehadiran.

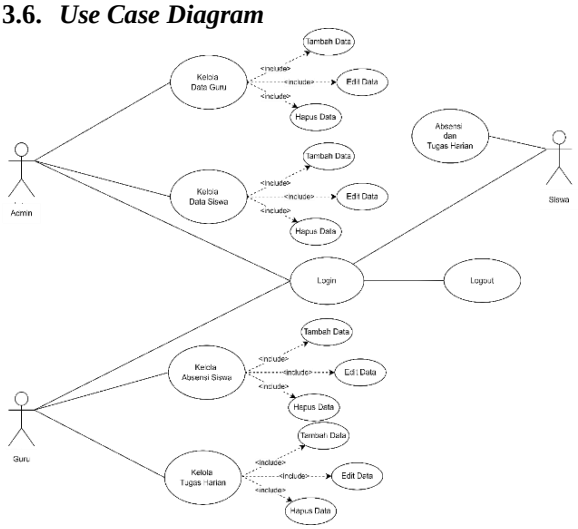
3.5. Flowchart Algoritma *Selection Sort*

Gambar 2. ditampilkan *Flowchart* algoritma *Selection Sort* yang menggambarkan proses pengurutan data. *Flowchart* ini dimulai dengan inisialisasi *array*, diikuti dengan iterasi untuk menentukan elemen terkecil dalam *array* yang belum terurut. Proses ini melibatkan penentuan indeks minimum (*min*) dalam setiap iterasi, pembandingan nilai elemen *array* dengan elemen minimum, dan pertukaran posisi elemen jika ditemukan nilai yang lebih kecil. Iterasi dilakukan hingga seluruh elemen *array* berada dalam urutan yang benar. *Flowchart* ini secara visual membantu memahami langkah-langkah algoritma *Selection Sort* yang digunakan untuk menyortir data secara efisien.



Gambar 2. Flowchart Algoritma *Selection Sort*

3.6. *Use Case Diagram*

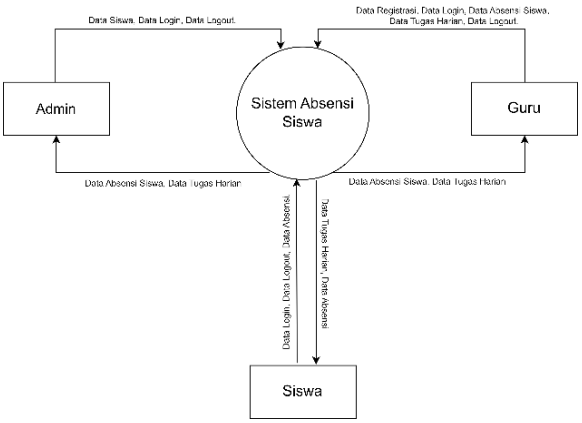


Gambar 3. *Use case diagram*

Gambar 3. ditampilkan *use case diagram* yang menggambarkan interaksi antara aktor dan sistem pada aplikasi yang dikembangkan, yang melibatkan tiga aktor utama, yaitu Admin, Guru, dan Siswa, dengan fungsionalitas khusus masing-masing. Admin bertanggung jawab mengelola data guru, data siswa, dan absensi siswa yang mencakup operasi tambah, edit, dan hapus data. Guru memiliki akses untuk mengelola absensi siswa serta tugas harian, juga dengan kemampuan untuk menambah, mengedit, dan menghapus data. Sementara itu, Siswa dapat melakukan *login*, *logout*, absensi, serta mengakses

tugas harian. Diagram ini menunjukkan relasi *include* pada setiap fungsi untuk menggambarkan hubungan inheren antar operasi, seperti pengelolaan data yang mencakup berbagai proses pendukung, sehingga memberikan gambaran menyeluruh tentang bagaimana sistem dirancang untuk memenuhi kebutuhan para pengguna.

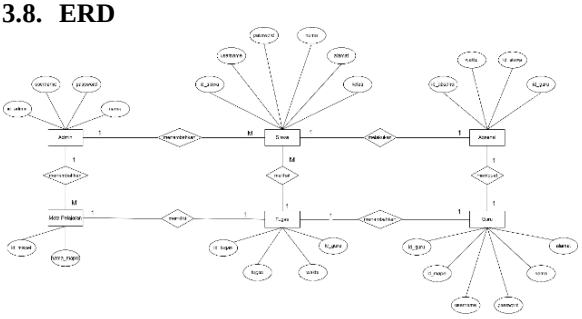
3.7. Diagram Konteks



Gambar 4. Diagram Konteks

Gambar 4. ditampilkan diagram konteks yang menggambarkan alur informasi antara sistem absensi siswa dengan tiga aktor utama, yaitu Admin, Guru, dan Siswa. Admin memiliki akses untuk melakukan *login*, input data siswa, dan *logout*, serta menerima laporan absensi siswa dan tugas harian. Guru berperan dalam mendaftarkan akun, *login*, menginput data absensi siswa, menginput tugas harian, dan *logout*, sekaligus mendapatkan laporan absensi siswa dan tugas harian. Sementara itu, Siswa dapat melakukan *login*, *logout*, melaksanakan absensi, serta melihat tugas harian. Diagram ini memberikan gambaran menyeluruh mengenai interaksi sistem dengan pengguna dan alur data yang mendukung fungsionalitas sistem.

3.8. ERD



Gambar 5. ERD

Pada gambar 5 ditampilkan *Entity Relationship Diagram* (ERD) yang menjelaskan hubungan antar entitas dalam sistem. Diagram ini menggambarkan entitas utama, seperti Admin, Siswa, Guru, Mata Pelajaran, Tugas, dan Absensi, beserta atribut-atribut yang dimiliki oleh masing-masing entitas. Hubungan antar entitas diilustrasikan melalui relasi, seperti Admin menambahkan data Siswa dan Mata Pelajaran, Guru membuat Tugas dan Absensi, serta Siswa melakukan Absensi dan melihat Tugas. Relasi-relasi tersebut disertai dengan kardinalitas untuk menunjukkan hubungan satu ke satu, satu ke banyak, atau banyak ke banyak, yang bertujuan untuk memvisualisasikan struktur data sistem secara detail.

3.9. Proses Perhitungan *Selection Sort*

Tahap implementasi algoritma *Selection Sort* dalam sistem absensi siswa di SMA Negeri 1 Nawangan Pacitan dilakukan untuk mengurutkan data kehadiran dari tingkat keaktifan tertinggi hingga terendah. Algoritma bekerja dengan mencari nilai tertinggi dari daftar yang belum diurutkan, menempatkannya di posisi pertama, lalu mengulang proses untuk posisi berikutnya hingga seluruh data terurut. Setiap iterasi membandingkan elemen, menemukan nilai tertinggi, dan menukar posisinya (*swap*) hingga seluruh data tersusun dengan benar. Tabel 1 menyajikan contoh 10 data siswa kelas X IPS 1 yang digunakan dalam proses pengurutan ini.

Tabel 1. Data siswa dna keaktifan siswa

No	Nama Siswa	Keaktifan
1	Ahmad Fadilah	75
2	Budi Santoso	85
3	Citra Maharani	60
4	Dian Prasetyo	90
5	Eka Wulandari	55
6	Fajar Kurniawan	80
7	Gita Lestari	70
8	Hendra Saputra	95
9	Intan Permatasari	65
10	Joko Prasetyo	88

Langkah-langkah *Selection Sort* proses pengurutan sebagai berikut :

a. Iterasi 1:
- Array Awal: [75, 85, 60, 90, 55, 80, 70, 95, 65, 88]
- Cari nilai minimum dari indeks 0-9 → 55 (indeks 4).
- Tukar 55 dengan elemen pertama (75).
- Hasil: [55, 85, 60, 90, 75, 80, 70, 95, 65, 88]

b. Iterasi 2:
- Array Awal: [55, 85, 60, 90, 75, 80, 70, 95, 65, 88]
- Cari nilai minimum dari indeks 1-9 → 60 (indeks 2).
- Tukar 60 dengan elemen kedua (85).
- Hasil: [55, 60, 85, 90, 75, 80, 70, 95, 65, 88]

c. Iterasi 3:
- Array Awal: [55, 60, 85, 90, 75, 80, 70, 95, 65, 88]
- Cari nilai minimum dari indeks 2-9 → 65 (indeks 8).
- Tukar 65 dengan elemen ketiga (85).
- Hasil: [55, 60, 65, 90, 75, 80, 70, 95, 85, 88]

d. Iterasi 4:
   - Array Awal: [55, 60, 65, 90, 75, 80, 70, 95, 85, 88]
   - Cari nilai minimum dari indeks 3-9 → 70 (indeks 6).
   - Tukar 70 dengan elemen keempat (90).
   - Hasil: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]

e. Iterasi 5:
   - Array Awal: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]
   - Cari nilai minimum dari indeks 4-9 → 75 (indeks 4).
   - Tidak ada perubahan karena sudah berada di posisi benar.
   - Hasil: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]

f. Iterasi 6:
   - Array Awal: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]
   - Cari nilai minimum dari indeks 5-9 → 80 (indeks 5).
   - Tidak ada perubahan karena sudah berada di posisi benar.
   - Hasil: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]

g. Iterasi 7:
   - Array Awal: [55, 60, 65, 70, 75, 80, 90, 95, 85, 88]
   - Cari nilai minimum dari indeks 6-9 → 85 (indeks 8).
   - Tukar 85 dengan elemen ketujuh (90).
   - Hasil: [55, 60, 65, 70, 75, 80, 85, 95, 90, 88]

h. Iterasi 8:
   - Array Awal: [55, 60, 65, 70, 75, 80, 85, 95, 90, 88]
   - Cari nilai minimum dari indeks 7-9 → 88 (indeks 9).
   - Tukar 88 dengan elemen kedelapan (95).
   - Hasil: [55, 60, 65, 70, 75, 80, 85, 88, 90, 95]

i. Iterasi 9:
   - Array Awal: [55, 60, 65, 70, 75, 80, 85, 88, 90, 95]
   - Cari nilai minimum dari indeks 8-9 → 90 (indeks 8).
   - Tidak ada perubahan karena sudah di posisi benar.

j. Hasil Akhir : [55, 60, 65, 70, 75, 80, 85, 88, 90, 95]

Setelah dilakukan proses pengurutan menggunakan algoritma *Selection Sort*, nilai keaktifan siswa berhasil diurutkan dari yang terendah hingga yang tertinggi. Nilai keaktifan terendah adalah 55, yang diperoleh oleh Eka Wulandari, sedangkan nilai keaktifan tertinggi adalah 95, yang dimiliki oleh Hendra Saputra. Proses ini dilakukan dengan cara mencari nilai terkecil di setiap iterasi dan menukarnya ke posisi yang benar, hingga seluruh data berada dalam urutan yang meningkat. Dengan demikian, hasil akhir menunjukkan bahwa Eka Wulandari memiliki tingkat keaktifan paling rendah di kelas, sementara Hendra Saputra menunjukkan tingkat keaktifan paling tinggi.

3.10. Pengujian

Setelah tahap implementasi, pengujian perangkat lunak dilakukan melalui metode *white box testing*, yang melibatkan pemeriksaan internal dan detail implementasi kode oleh pengembang atau pihak yang memiliki akses ke struktur internal perangkat lunak. Fokus utama dalam metode ini adalah memastikan bahwa setiap komponen dan alur logika dalam perangkat lunak berfungsi dengan benar sesuai spesifikasi yang telah ditetapkan. Pengujian mencakup analisis struktur *kode, coverage testing*, dan pemeriksaan logika internal untuk mengidentifikasi potensi bug atau kesalahan. Jika ditemukan masalah, pengembang dapat langsung melakukan perbaikan. Dengan menggunakan *white box testing*, perangkat lunak diuji secara lebih mendalam dari perspektif internal, memastikan kualitas dan keandalan dari segi struktur dan implementasi kode, sesuai dengan pendekatannya.

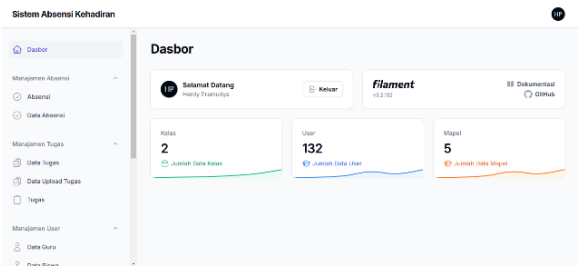
4. HASIL DAN PEMBAHASAN

Hasil pada penelitian ini mencakup implementasi *interface* sistem dan pengujian *whitebox testing*, yang dilakukan untuk memastikan logika internal sistem berjalan sesuai dengan yang diharapkan. Dalam *whitebox testing*, pengujian difokuskan pada struktur kode dan alur algoritma, memastikan bahwa setiap bagian kode berfungsi dengan benar dan bebas dari bug. Pengujian ini juga mencakup pengujian unit untuk memastikan bahwa setiap fungsi dalam sistem bekerja secara terpisah sebelum diintegrasikan.

4.1. Implementasi *Interface*

Pada tahap implementasi, dua komponen utama yang digunakan adalah *interface* sistem. *Interface* sistem memungkinkan pengguna untuk memasukkan data kehadiran siswa dan menampilkan hasil pengurutan keaktifan. Algoritma *Selection Sort* mengurutkan data kehadiran dengan memilih elemen terkecil dari daftar yang belum diurutkan dan menukarnya ke posisi yang sesuai, menghasilkan data yang terstruktur dan mudah dipahami.

4.2. *Dashboard*

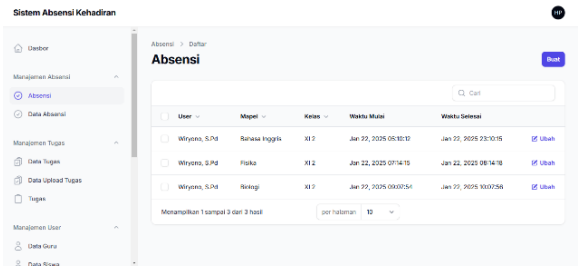


Gambar 6. *Dashboard*

Gambar 6 menampilkan *dashboard* sistem absensi siswa untuk pengurutan keaktifan berdasarkan kehadiran. *Dashboard* ini memungkinkan admin untuk melihat informasi penting seperti jumlah kelas, user, dan mata pelajaran, serta mengelola absensi,

tugas, dan data pengguna dengan tampilan yang intuitif.

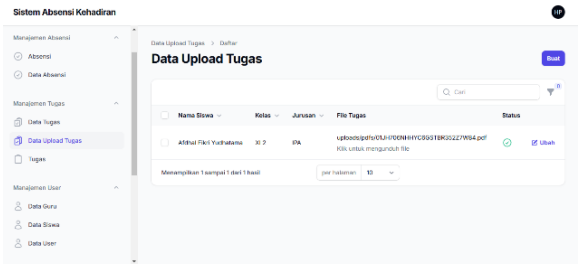
4.3. Data Absensi



Gambar 7. Absensi

Gambar 7 menampilkan data absensi dalam sistem, mencakup informasi seperti nama pengguna, mata pelajaran, kelas, dan waktu. Admin dapat mencari, menambah, atau mengubah data absensi melalui tombol "Ubah", mempermudah pengelolaan dan pemantauan kehadiran siswa.

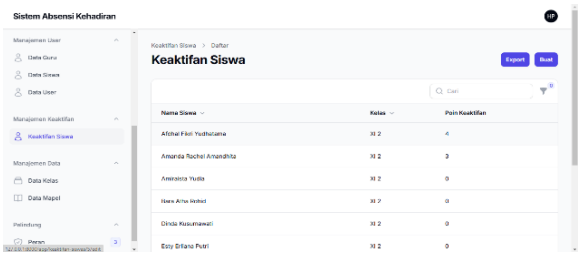
4.4. Data *Upload* Tugas



Gambar 8. Data *Upload* Tugas

Gambar 8 menampilkan data upload tugas, mencakup informasi guru, jurusan, kelas, dan waktu. Admin atau guru dapat menambah, mencari, atau mengedit tugas melalui tombol "Ubah", mempermudah pengelolaan tugas siswa.

4.5. Keaktifan Siswa



Gambar 9. Keaktifan Siswa

Gambar 9 menunjukkan data keaktifan siswa dalam sistem absensi kehadiran, yang memungkinkan pengelolaan, pemantauan, dan pengurutan keaktifan siswa berdasarkan poin kehadiran untuk setiap kelas yang terdaftar.

4.6. Pengujian Sistem

Pengujian sistem adalah tahap pengujian perangkat lunak di mana seluruh sistem atau aplikasi diuji sebagai satu kesatuan yang terintegrasi. Tujuan dari pengujian sistem adalah untuk memastikan bahwa semua komponen dan modul dari perangkat lunak berfungsi dengan baik bersama-sama dan memenuhi spesifikasi serta kebutuhan yang telah ditentukan. Selain pengujian fungsional, *whitebox testing* juga dilakukan untuk memastikan bahwa alur logika dan struktur internal sistem berjalan sesuai yang diharapkan. *Whitebox testing* memeriksa kode sumber sistem secara rinci untuk mendeteksi potensi kesalahan atau bug pada bagian-bagian yang tidak terdeteksi dalam pengujian fungsional. Pengujian ini mencakup analisis jalur, percabangan, dan pengujian unit untuk memastikan keandalan dan ketahanan sistem dalam menangani berbagai skenario *input*.

4.7. Pengujian *Unit*

Pengujian unit pada *method* "sortByPoinKeaktifan" melibatkan beberapa langkah. Pertama, data siswa dengan atribut poin keaktifan disiapkan dalam bentuk *array* objek *Student*. Selanjutnya, metode ini akan mengurutkan data berdasarkan poin keaktifan dari nilai tertinggi ke terendah menggunakan algoritma *Selection Sort*. Pengujian dilakukan dengan beberapa skenario, yaitu ketika data sudah terurut, data acak, dan data dengan nilai yang sama. Jika sorting berhasil, maka output yang diharapkan adalah urutan siswa berdasarkan poin keaktifan dalam format yang benar. Jika *sorting* gagal, pengujian akan menangkap kesalahan dalam implementasi, seperti kesalahan logika dalam membandingkan nilai atau dalam pertukaran elemen. Dengan pengujian ini, diharapkan semua jalur eksekusi dalam *method* "sortByPoinKeaktifan" telah diuji dengan baik dan memberikan hasil yang sesuai dengan kebutuhan sistem.

Tabel 1. *Pseudocode Method "function sortByPoinKeaktifan()"* dari *Class"Student"*

Pseudocode	***Node***
$records = $records->values(); $n = $records->count();	1 1

for ($i = 0; $i < $n - 1; $i++) {
$minIndex = $i;$minIndex = $i;
for ($j = $i + 1; $j < $n; $j++) {
if ($records[$j]->getPoinKeaktifanAttribute()
> $records[$maxIdx]-
>getPoinKeaktifanAttribute()) {
$maxIdx = $j;
}
}
	2
	2
	2
	2
	3
	3
	3
	3
	3
	3
	3
	3

if ($maxIdx !== $i) {
$temp = $records[$i];
$records[$i] = $records[$maxIdx];
$records[$maxIdx] = $temp;
}
}
	4
	4
	4
	4
	4
	4

return $records; }	5 5

4.8. *Flowgraph*

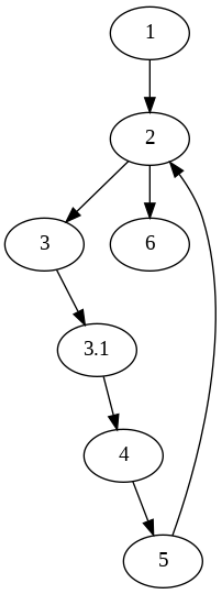
Flowgraph ini merupakan representasi visual dari semua jalur logika dalam *method sortByPoinKeaktifan* dari class *Student*. Diagram ini terdiri dari *Node* dan *edges*, di mana setiap *Node* mewakili satu atau lebih pernyataan dalam kode, sedangkan setiap *edge* menunjukkan aliran kontrol dari satu pernyataan ke pernyataan berikutnya. Dalam pengujian *white box* menggunakan teknik *basis path testing*, flow graph berfungsi untuk mengidentifikasi semua jalur independen yang perlu diuji agar memastikan cakupan lengkap dari kondisi logis dan kontrol dalam algoritma *Selection Sort*. Setiap jalur diuji untuk memastikan bahwa proses pengurutan daftar *records* berdasarkan *Poin Keaktifan* dalam urutan menurun berjalan dengan benar. Dengan demikian, teknik ini membantu mendeteksi potensi kesalahan dalam algoritma dan memastikan bahwa setiap jalur eksekusi telah diuji secara menyeluruh.

Cyclomatic Complexity (V(G)) dihitung untuk mengukur kompleksitas kode berdasarkan *flowgraph*. *Menggunakan* rumus V(G) = E - N + 2 dengan E = 9 dan N = 8, diperoleh V(G) = 3. Perhitungan alternatif V(G) = P + 1 dengan P = 2 juga menghasilkan V(G) = 3.

Hasil ini menunjukkan bahwa terdapat tiga jalur independen yang harus diuji untuk memastikan cakupan penuh dan validasi logika sistem.

Dengan *Cyclomatic Complexity* (V(G)) = 3, terdapat tiga jalur independen:

- *Node* 1 → 2 → 6 → 5 (jalur utama tanpa percabangan).
- *Node* 1 → 2 → 3 → 3.1 → 4 → 5 (percabangan melalui *Node* 3).
- *Node* 1 → 2 → 6 → kembali ke 2 → 3 → 3.1 → 4 → 5 (jalur dengan *loop*).



Gambar 10. *Flowgraph Method function "sortByPoinKeaktifan()"* dari *Class"Studens"*

Berdasarkan pengujian *whitebox* terhadap *method sortByPoinKeaktifan*, serta analisis dari Tabel.1 (*pseudocode*) dan Gambar 16 (*flowgraph*), dapat disimpulkan bahwa algoritma Selection Sort berhasil diimplementasikan untuk mengurutkan daftar siswa berdasarkan Poin Keaktifan dalam urutan menurun. Algoritma *Selection Sort* berhasil mengurutkan siswa berdasarkan Poin Keaktifan dalam urutan menurun. Dengan *Cyclomatic Complexity* (V(G)) = 3, pengujian mencakup eksekusi langsung, iterasi pencarian nilai maksimum, dan iterasi berulang sebelum kondisi akhir. Basis path testing memastikan seluruh jalur logika telah diverifikasi, menunjukkan bahwa metode ini berjalan sesuai harapan, memiliki kompleksitas rendah, serta mudah diuji dan di-debug, sehingga dapat disimpulkan bahwa *sortByPoinKeaktifan* telah terverifikasi dengan baik dan andal.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil implementasi algoritma *selection sort* dalam Sistem Absensi Siswa SMA Negeri 1 Nawangan, dapat disimpulkan bahwa algoritma ini mampu diintegrasikan secara efektif untuk mengurutkan data absensi siswa berdasarkan poin keaktifan dengan tingkat efisiensi yang optimal. Proses pengurutan dilakukan secara sistematis, dimulai dengan memilih elemen terkecil dari daftar data yang belum terurut, kemudian menempatkannya di posisi pertama daftar terurut. Langkah ini diulangi hingga seluruh data tersusun dengan benar. Hasil implementasi menunjukkan bahwa algoritma ini bekerja secara akurat dan konsisten dalam mengelompokkan data absensi, memudahkan pengelolaan informasi berdasarkan kriteria tertentu, seperti nama siswa, kelas, atau poin keaktifan. Algoritma *Selection Sort* juga memiliki struktur logika yang sederhana, yang mempermudah proses *debugging* dan pengujian sistem. Hal ini memastikan keandalan sistem absensi dalam memberikan informasi yang akurat dan terorganisir. Dengan demikian, penggunaan Algoritma *Selection Sort* memberikan kontribusi yang signifikan terhadap efisiensi dan kualitas pengelolaan data absensi siswa.

Hasil pengujian *whitebox* pada tabel 1 dan gambar 4.23 menunjukkan bahwa algoritma ini mampu menangani dataset absensi dengan akurasi tinggi, menghasilkan urutan data yang konsisten sesuai kriteria yang ditentukan, seperti nama, kelas, atau poin keaktifan siswa. Selain itu, perhitungan *Cyclomatic Complexity* sebesar 3 menunjukkan bahwa algoritma memiliki struktur kontrol yang sederhana dan mudah dipahami, sehingga mempermudah proses debugging, optimasi, serta pemeliharaan sistem. Dengan jalur independen yang jelas dan terstruktur, pengujian unit memastikan bahwa seluruh jalur logis telah diuji secara menyeluruh, sehingga keandalan dan akurasi sistem meningkat. Penerapan Algoritma *Selection Sort* dalam sistem ini memberikan solusi yang efektif untuk

pengelolaan data absensi siswa yang lebih terorganisir, akurat, dan efisien.

Sebagai langkah pengembangan, sistem absensi siswa dapat ditingkatkan dengan menambahkan fitur analitik untuk visualisasi tren kehadiran, memungkinkan pihak sekolah memperoleh wawasan lebih mendalam. Selain itu, optimalisasi skalabilitas diperlukan agar sistem mampu menangani data dalam jumlah besar atau diterapkan di beberapa sekolah dengan dukungan *cloud*. Aspek keamanan juga perlu diperkuat melalui enkripsi dan autentikasi yang lebih canggih guna melindungi data absensi dari kebocoran atau akses tidak sah.

DAFTAR PUSTAKA

[1] J. Karaman, P. M. Gunawan, S. Firdhossiah, L. M. M. Fitriani, Sucipto, and R. Indriati, "Rancang Bangun Sistem Absensi Berbasis Website di SMK Muhammadiyah 3 Dolopo," *J. Comput. Sci. Inf. Technol.*, vol. 4, no. 1, pp. 1–15, 2024, [Online]. Available: https://journal.fkpt.org/index.php/Explorer/article/view/818

[2] J. Karaman *et al.*, "Sosialisasi dan Pendampingan Tenaga Pendidik dalam Penerapan Teknologi Absensi Berbasis Qr Code di Sekolah Dasar Desa Tugu," *JMM - J. Masy. Merdeka*, vol. 6, no. 2, pp. 102–113, 2024, doi: 10.51213/jmm.v6i2.143.

[3] M. A. R. Sikumbang, R. Habibi, and S. F. Pane, "Sistem Informasi Absensi Pegawai Menggunakan Metode RAD dan Metode LBS Pada Koordinat Absensi," *J. Media Inform. Budidarma*, vol. 4, no. 1, pp. 59–65, 2020, doi: 10.30865/mib.v4i1.1445.

[4] S. Hanifa, K. Khusaini, and A. Widiarti, "Kehadiran Berkontribusi dalam Peningkatan Kinerja Akademik Mahasiswa," *SAP (Susunan Artik. Pendidikan)*, vol. 8, no. 3, p. 392, 2024, doi: 10.30998/sap.v8i3.20399.

[5] A. G. Mulia, "Sistem Informasi Absensi berbasis WEB di Politeknik Negeri Padang," *J. Teknol. Inf. Indones.*, vol. 5, no. 1, pp. 11–17, 2020, doi: 10.30869/jtii.v5i1.519.

[6] J. B.Hayfron-Acquah, O. Appiah, and K. Riverson, "Improved Selection Sort Algorithm," *Int. J. Comput. Appl.*, vol. 110, no. 5, pp. 29–33, 2020, doi: 10.5120/19314-0774.

[7] A. Syahputra, "Implementasi Algoritma Selection Sort Untuk Pengurutan Nilai IPK Mahasiswa Universitas Potensi Utama," *J. Tek. Inform. Kaputama*, vol. 6, no. 2, pp. 390–398, 2022.

[8] Yunita Wisda Tumarta Arif, "Implementasi Metode Selection Sort Pada Sistem Informasi Retensi Dokumen Rekam Medis Di Klinik Pku Muhammadiyah Karanganom, Klaten," *Elkom J. Elektron. dan Komput.*, vol. 15, no. 1, pp. 108–117, 2022, doi: 10.51903/elkom.v15i1.789.

[9] K. Priambodo and J. Sasongko Wibowo, "Implementasi Algoritma Selection Sort Untuk Perangkingan Poin Pada E-Sports Tournament Garuda League," *Proceeding SENDIU 2021*, pp. 978–979, 2021, [Online]. Available: www.garudaleague.com

[10] E. Sunandar, "Perbandingan Metode Selection Sort dan Insertion Sort Dalam Pengurutan Data Menggunakan Bahasa Program Java," *Petir*, vol. 12, no. 2, pp. 172–178, 2019, doi: 10.33322/petir.v12i2.485.

[11] E. P. Nimasari, A. F. Cobantoro, S. D. Andika, and M. B. Setyawan, "Analisis Implementasi Teknologi Pembelajaran di Bebas UMPO," *J. Ilm. Edutic Pendidik. dan Inform.*, vol. 9, no. 2, pp. 162–177, 2023, doi: 10.21107/edutic.v9i2.19938.

[12] I. Abdurrozzaq, Y. Litanianda, and A. Fajaryanto, "Implementasi Decision Support System Rekomendasi Beasiswa Pendidikan Menggunakan Metode Simple Additive Weighting," *J. Sains Komput. Dan Sist. Inf.*, vol. 2, no. 1, pp. 142–149, 2024.

[13] N. Anggraini, A. F. Cobantoro, and F. Masykur, "Pemberian Prioritas Penambahan Guru Sekolah Menengah Atas Di Kabupaten Magetan Dengan Metode Weighted Sum Model," *Komputek*, vol. 6, no. 1, pp. 106–118, 2022.

[14] A. Syaebani, D. V. Tyasmala, R. Maulani, E. D. Utami, and S. N. Wahyuni, "Pengembangan Sistem Informasi Pelayanan Surat Menyurat (SIRA) Berbasis Website Dengan Menggunakan Framework Codeigniter," *J. Inf. Syst. Manag.*, vol. 3, no. 2, pp. 59–65, 2021, doi: 10.24076/joism.2021v3i2.446.

[15] J. Banjarnahor, D. Bawamenewi, C. Tanoto, and M. NK Nababan, "Implementasi Metode Selection Sort Dalam Sistem Repository Skripsi," *J. Sist. Inf. dan Ilmu Komput. Prima(JUSIKOM PRIMA)*, vol. 5, no. 2, pp. 107–113, 2022, doi: 10.34012/jurnalsisteminformasidanilmukomputer.v5i2.2389.

[16] M. F. Londjo, "Implementasi White Box Testing Dengan Teknik Basis Path Pada Pengujian Form Login," *J. Siliwaangi*, vol. 7, no. 2, pp. 35–40, 2021.

[17] A. C. Praniffa, A. Syahri, F. Sandes, U. Fariha, Q. A. Giansyah, and M. L. Hamzah, "Pengujian Black Box Dan White Box Sistem Informasi Parkir Berbasis Web Black Box and White Box Testing of Web-Based Parking Information System," *J. Test. dan Implementasi Sist. Inf.*, vol. 1, no. 1, pp. 1–16, 2023.

[18] K. N. Arditya, M. B. Setyawan, and A. F. Cobantoro, "Integrasi Algoritma Fisher-Yates Sebagai Pengembangan E-Learning Di Unit Kegiatan Belajar Mandiri," *Komputek*, vol. 5, no. 1, p. 58, 2021, doi: 10.24269/jkt.v5i1.684.

[19] W. W. Royce, *Managing the Development of Large Software Systems*, no. August. 1970. doi: 10.7551/mitpress/12274.003.0035.