Jurnal Ilmiah Sistem Informasi dan Ilmu Komputer
Volume 5, Nomor 2, Juli 2025
p-ISSN: 2827-8135; e-ISSN: 2827-7953, Hal. 18-35
DOI: https://doi.org/10.55606/juisik.v5i2.1073
Available Online at: https://journal.sinov.id/index.php/juisik

Pengujian Sistem Rekomendasi Penilaian Pengemudi Transportasi Umum Nasional Menggunakan *Path Testing* pada *White Box Testing*

Safrizal
Universitas Pembangunan Jaya, Indonesia

Alamat: Jl. Cendrawasih Raya Blok B7/P Bintaro Jaya, Sawah Baru, Ciputat, Tangerang Selatan 15413
*Koresprodensi Penulis : safrizal.abdurrahman@upj.ac.id**

Abstract*. Driver assessment recommendation system to conduct assessment and recommendations of national public transportation driver competency and behavior. To ensure the reliability and stability of the program logic in this system, testing is carried out using the White Box Testing approach, especially the Path Testing method. This method tests each execution path in the program control structure to identify potential logic errors and ensure that each condition has been thoroughly tested. Testing is carried out by calculating cyclomatic complexity, describing the program flowchart, and identifying and tracing independent paths. The test results show that all logic paths have run as expected without any functional defects. This indicates that the system has a strong logic structure and is ready to be integrated into a national-scale operational environment. Thus, Path Testing has proven effective in ensuring system quality in terms of internal structure.*

Keywords*: White Box Testing, Path Testing, Recommendation System, Driver Assessment, Cyclomatic Complexity.*

Abstrak. Sistem rekomendasi penilaian pengemudi untuk melakukan penilaian dan rekomendasi kompetensi dan perilaku pengemudi transportasi umum nasional. Untuk memastikan keandalan dan stabilitas logika program dalam sistem ini, dilakukan pengujian menggunakan pendekatan White Box Testing, khususnya metode Path Testing. Metode ini menguji setiap jalur eksekusi dalam struktur kendali program guna mengidentifikasi potensi kesalahan logika dan memastikan setiap kondisi telah diuji secara menyeluruh. Pengujian dilakukan dengan menghitung kompleksitas siklomatik, menggambarkan flowchart program, serta mengidentifikasi dan menelusuri jalur independen. Hasil pengujian menunjukkan bahwa seluruh jalur logika telah berjalan sesuai dengan yang diharapkan tanpa ditemukan cacat fungsional. Hal ini mengindikasikan bahwa sistem memiliki struktur logika yang kuat dan siap untuk diintegrasikan dalam lingkungan operasional berskala nasional. Dengan demikian, Path Testing terbukti efektif dalam menjamin kualitas sistem dari sisi struktur internal.

Kata kunci: White Box Testing, Path Testing, Sistem Rekomendasi, Penilaian Pengemudi, Kompleksitas Siklomatik.

1. LATAR BELAKANG

Penelitian ini digunakan untuk melakukan penilaian terhadap perilaku dan kompetensi pengemudi transportasi umum, seperti sopir bus dan angkot, di tingkat nasional. Penilaian Perilaku dan kompetensi pengemudi angkutan umum diadopsi menggunakan kriteria yang ada pada Standar Kompetensi Lulusan (SKL) berbasis Kerangka Kualifikasi Nasional Indonesia (KKNI) mengenai pengemudi pemula level II. Instrumen penilaian berdasarkan Standar Kompetensi Lulusan (SKL). SKL memiliki 4 komponen kompetensi, 4 komponen kompetensi tersebut antara lain adalah sikap dan tata nilai, kemampuan di bidang kerja, pengetahuan yang dikuasai, hak dan tanggung jawab.(Peraturan Pemerintah No.12, tahun 2012). Pada penelitian ini menggunakan pengujian White Box testing yaitu Path Testing. Metodologinya adalah dengan cara melakukan Analisis Control Flow Graph (CFG) dengan menyusun CFG dari kode

Received: Mei 03, 2025; Revised: Mei 15, 2025; Accepted: Mei 29, 2025; Online Available: Mei 31, 2025;

sumber tiap fungsi utama, menganalisis alur logika program seperti percabangan, pengulangan, dan kondisi. Penentuan Basis Path dan Jalur Independen dengan menghitung Cyclomatic Complexity dan menyusun test case berdasarkan jalur logika yang telah ditentukan.(Madhavi, 2016). Kekurangan dari pengujian Path testing pada White Box yaitu perlu pemahaman mendalam tentang struktur internal kode dan logika program. Menguji jalur yang ada dalam kode, jalur logika yang seharusnya ada namun tidak akan terdeteksi jika tidak diimplementasikan (Katlon,2025). Kelebihan dari path testing pada pengujian White Box adalah mendeteksi kesalahan logika secara menyeluruh, memeriksa semua jalur eksekusi, sehingga dapat menemukan kesalahan logika yang tidak terdeteksi oleh metode lain. (Pressman & Maxim, 2020). Cakupan pengujian yang tinggi terhadap struktur kontrol program Path testing memaksa penguji untuk menguji semua struktur percabangan (if, case, loop), sehingga meningkatkan jaminan kualitas kode (Sommerville, 2011).

Mengidentifikasi jalur yang tidak pernah dieksekusi (dead code)

Dengan analisis jalur, dapat ditemukan bagian program yang tidak memiliki kontribusi terhadap output.(Ali & Saleem, 2023). Meningkatkan pemahaman terhadap desain algoritma dan logika program Karena membutuhkan pemahaman struktur dalam kode, path testing membantu pengembang memahami logika sistem secara mendalam (Bardin et al., 2014). Bisa diotomatisasi dengan alat bantu pengujian modern Banyak tools pengujian yang mendukung path testing untuk mengidentifikasi jalur kompleks secara otomatis.(Jorgensen, 2018).

Permasalahan Pengujian White Box Testing dalam Penerapan Path Testing pada Sistem Penilaian Perilaku dan Kompetensi Pengemudi Transportasi Umum Nasional adalah kesulitan mengidentifikasi Jalur Independen Path testing mengharuskan identifikasi basis path atau jalur independen melalui control flow graph (CFG). (IEEE Standard for Software and System Test Documentation, 2008). Permasalahan pada penelitian ini adalah kesulitan mengidentifikasi Jalur Independen dalam white box testing, khususnya path testing. Terdapat banyak kondisi bercabang (decision points) dan loop (pengulangan), Hal ini menghasilkan CFG yang sangat kompleks, sehingga sulit untuk menentukan jalur independen secara manual(Kaner, 2001). Gap analys pada penelitian ini adalah banyak penelitian hanya menggunakan Black Box Testing atau pengujian fungsional biasa untuk sistem transportasi, Pengujian sistem transportasi seringkali fokus pada hasil input-output tanpa menguji struktur logika internal program. Kurangnya penerapan White Box Testing berbasis Path Testing dalam sistem penilaian perilaku pengemudi. (Ammann & Offutt, 2016).

Belum banyak penelitian di Indonesia yang menerapkan metode formal pengujian seperti path testing pada sistem transportasi(Jorgensen, 2018). Keterbaruan (Novelty)

Penelitian Penerapan Path Testing pada Sistem Penilaian Perilaku Pengemudi Kebanyakan penelitian white box testing dengan pendekatan *path testing* diterapkan pada aplikasi sederhana seperti form login, aplikasi laundry, dan sistem kasir. Penelitian ini menggunakan perhitungan *Cyclomatic Complexity* untuk menentukan jumlah jalur independen dan menguji semuanya secara menyeluruh — hal yang jarang dibahas dalam implementasi testing sistem perilaku manusia.(Katlon, 2025).

2. KAJIAN TEORITIS

Pengujian White box dilakukan pada tingkat unit, integrasi, dan sistem untuk mendeteksi kesalahan logika, debugging, dan memvalidasi asumsi pemrograman. Metode ini mencakup pengujian menyeluruh terhadap semua jalur kode, membantu optimasi, dan memberikan pedoman penghentian pengujian. (Kumar et al., 2015) .*Path Testing* adalah salah satu satu teknik dalam white box testing yang fokus pada pengujian semua jalur eksekusi logis dalam kode program. Ini akan menghasilkan sistem yang lebih stabil, akurat, dan dapat diandalkan. (Gusdevi et al., 2022) .White box testing metode pengujian perangkat lunak yang melibatkan pemahaman mendalam terhadap struktur internal program. Salah satu teknik yang digunakan dalam White Box Testing adalah Basic Path Testing, yang fokus pada pengujian setiap jalur eksekusi dalam program. Pengujian ini bertujuan untuk memastikan bahwa program berfungsi dengan benar dan dapat mengelola proses peminjaman dengan baik (Dimas Saputro & Azzahra Narwastika, 2023) Pengujian white-box dilakukan dengan menganalisis *source code* aplikasi terlebih dahulu, menentukan file dan fungsi yang harus diuji, serta membuat flow graph dari kode sumber (Nurwicaksono et al., 2023) Pengujian yang dihasilkan dari white box testing bergantung pada flowchart yang ditransformasikan menjadi flowgraph. Metode white box testing dengan teknik basic path menghasilkan tes uji yang semakin besar seiring dengan ukuran sistem yang diuji (Rini & Kusmaya Putri, 2023.) White box testing disebut juga sebagai pengujian struktural, di mana penguji memiliki akses ke seluruh struktur atau logika perangkat lunak yang diuji (Rini & Kusmaya Putri, 2023.) Langkah langkah yang digunakan untuk pengujian jalur adalah Analisa flowchart/ Sourcode, buat Flow Graph, Cylomatic Complexity, Penentuan Jalur Indepen, Perancangan Test case. *Cyclomatic Complexity* adalah mengukur jumlah jalur eksekusi berbeda dalam suatu program. Semakin tinggi angkanya, semakin kompleks kode tersebut, dan semakin sulit untuk diuji dan dipelihara. V(G) = E – N + 2. Keterangan: E = jumlah edges pada flowgraph N = jumlah nodes pada flowgraph, P = jumlah *predicates nodes* pada flowgraph (Peñalosa et al., 2019) Penelitian ini menggunakan pengujian White Box, yaitu pengujian jalur independen, pengembangan grafik aliran, perhitungan

kompleksitas siklomatik, dan pengembangan matriks grafik.(Syaikhuddin et al., 2018). Metode white box testing mengutamakan pengetahuan akan struktur sistem. Dalam praktik nyata, metode ini dapat diterapkan dalam pengujian produk sistem informasi untuk memastikan kehandalan dan kualitasnya (Mega & Safrizal,2025). Penelitian dengan mengambil studi kasus pada sistem informasi inventori stok barang pada suatu usaha jasa pengiriman barang. Teknik pengujian sistem informasi ini menggunakan metode black box testing, dengan teknik pengujian *Equivalence Partitioning*.(Rahman Abdillah et al., 2024). White box testing membutuhkan pemahaman mendalam tentang struktur internal perangkat lunak dan bahasa pemrograman yang digunakan. Hal ini menjadikannya kurang cocok bagi tim pengujian yang tidak memiliki pengetahuan teknis yang memadai atau akses ke kode sumber. Pengujian White Box berfungsi sebagai Pengujian Clear Box atau Pengujian Struktural untuk melakukan penilaian struktur internal aplikasi dan fungsi logis serta kode pemrograman. (Safrizal et al.,2024)

Kekuatan Pengujian White-Box dapat dipastikan semua jalur kode, cabang, loop, dan pernyataan kondisional telah tercakup sebagai identifikasi untuk kesalahan tersembunyi. Membantu mendeteksi Bug Dini dalam Kode untuk menemukan bug dan kerentanan keamanan di awal kode, yang tidak dapat dilakukan dalam pengujian black-box.(Li et al., 2009). Pengujian aplikasi web adalah inkan untuk menemukan bug kapan saja, sebelum rilis atau setiap hari.(Golian et al., 2022). Permasalahan pengembangan perangkat lunak ini adalah sering kali kurangnya pengujian yang menyebabkan kegagalan perangkat lunak. Untuk mempertahankan produk berkualitas tinggi dalam kondisi kinerja yang sangat baik, pengujian menjadi penting. Perangkat lunak dapat diuji dengan menggunakan teknik pengujian White Box, Black Box, atau Gray.(Izzat & Saleem, 2023).

Penelitian ini bertujuan untuk menghasilkan suatu metode optimasi pengujian white box pada program Java dengan memanfaatkan struktur percabangan dan pengulangan menggunakan metode basis path (Andrian Ibrahim & Saktian Laksito, 2024). Metode pengujian White box disesuaikan untuk model NLP berbasis transformator. Metode tersebut mencakup Mask Neuron Coverage (Mncover) yang mengukur seberapa menyeluruh lapisan perhatian dalam model dilaksanakan selama pengujian.(Sekhon et al., 2022). Pengujian White box berfokus pada struktur kontrol dalam program, untuk memverifikasi semua pernyataan dalam program dieksekusi setidaknya satu kali serta semua kondisi logis dijalankan. Pengujian dilakukan untuk memverifikasi persyaratan fungsional. Ini mencakup partisi ekuivalen, analisis nilai batas, pengujian berbasis kesalahan, pengujian acak, pengujian partisi, teknik grafik sebab akibat.(Chawan et al., 2012). Penelitian ini bertujuan untuk melengkapi algoritma

ini dengan fitur AI yang dapat dijelaskan (XAI) guna meningkatkan transparansi. Temuan pada penelitian ini menyoroti jalur yang menjanjikan untuk integrasi kinerja algoritma kotak hitam dengan kebutuhan akan transparansi dalam domain pengambilan keputusan yang kritis(Žlahtič et al., 2024). Pengujian perangkat lunak merupakan istilah penting untuk keandalan perangkat lunak. Pengujian memberikan struktur dan validitas asli pada perangkat lunak untuk kinerja yang efisien dalam kondisi operasional.(Sheakh et al., 2015). Dalam penelitian ini FRAMA-C/LTEST, perangkat generik dan terintegrasi untuk pengujian white-box otomatis program C. LTEST menyediakan dukungan terpadu untuk berbagai kriteria pengujian serta integrasi kriteria baru yang mudah. (Bardin et al., 2014)

Pengujian White box evolusioner merupakan pendekatan yang menjanjikan untuk otomatisasi lengkap pembuatan kasus uji yang berorientasi pada struktur. Pengujian evolusioner tidak dapat menemukan data uji yang valid. Hal ini dapat mengarah pada peningkatan efisiensi dan efektivitas pengujian White boxevolusioner.(Lammermann & Wappler, 2005.). Pengujian White box evolusioner sejauh ini belum dapat membuat pernyataan tentang keberadaan sasaran pengujian yang tidak dapat dicapai, dapat ditingkatkan dengan bantuan pengukuran perangkat lunak evolusioner. (Caniço & Santos, 2023). Penelitian ini menunjukkan metode ini berhasil mengidentifikasi dan memperbaiki kesalahan dalam logika pemrograman serta kerentanan sistem, yang berkontribusi pada peningkatan performa sebesar 20%, peningkatan stabilitas, dan pengurangan risiko terhadap ancaman keamanan.(Suryanta et al., 2025). Pengujian white box dengan teknik basis path diterapkan pada situs web sistem administrasi kependudukan, yaitu aplikasi komputer untuk melakukan administrasi kependudukan. Diperoleh bahwa path dari setiap fitur yang telah dilalui telah sesuai di antara hasil Cyclomatic Complexity dan Flowgraph yang dibuat.(Sasmito, 2020b). Pengujian ini menjelaskan langkah-langkah untuk melakukan pengujian aliran data serta cara merancang rangkaian pengujian yang memperhitungkan anomali. Pendekatan ini mencakup desain berbasis node, cakupan penemuan tren, perbandingan aplikasi web, dan pengujian analitis.(Ali & Saleem, 2023).

3. METODE PENELITIAN

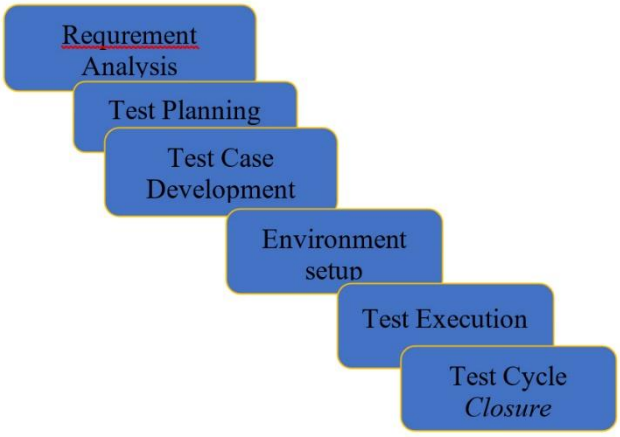
Pengujian Sistem Rekomendasi Penilaian Pengemudi Transportasi Umum Nasional Menggunakan Path Testing pada White Box Testing diterapkan pada Model Penilaian Sistem Perilaku dan Kompetensi Pengemudi Transportasi Umum Nasional. Dari Model Penilaian Sistem Perilaku dan Kompetensi Pengemudi Transportasi Umum Nasional dibuatlah Aplikasi penilaian Sistem Rekomendasi Penilaian Pengemudi Transportasi Umum Nasional

menggunakan Path Testing pada White Box Testing. Dari Aplikasi ini dilakukan Pengujian Path testing pada white Box testing. Metodologi penelitian yang digunakan adalah menggunakan *Software Testing Life Cycle (STLC)* dan Tahapan Pengujian White Box testing. Mengenai STLC dan Tahapan pengujian white box testing dijelaskan sebagai berikut: *Software Testing Life Cycle (STLC)* adalah sebuah proses sistematis yang terdiri dari beberapa tahapan yang dilakukan untuk memastikan kualitas dan keandalan perangkat lunak melalui pengujian yang terstruktur.

STLC adalah serangkaian tahapan yang dilakukan dalam proses pengujian perangkat lunak, dimulai dari perencanaan pengujian hingga penutupan pengujian, dengan tujuan menemukan dan memperbaiki bug, serta memastikan perangkat lunak bekerja sesuai spesifikasi yang diharapkan.

Tahapan dalam STLC:

a. Requirement Analysis (Analisis Kebutuhan)
    - Tim QA (Quality Assurance) menganalisis dokumen kebutuhan untuk memahami apa yang harus diuji.
    - Tujuan: mengidentifikasi kebutuhan pengujian dan kelayakan pengujian.
b. Test Planning (Perencanaan Pengujian)
    - Menyusun strategi pengujian, menentukan sumber daya, anggaran, jadwal, dan alat yang akan digunakan.
    - Output utama: *Test Plan* atau *Test Strategy Document*.
c. Test Case Development (Pengembangan Kasus Uji)
    - Menulis *test cases* dan *test scripts* berdasarkan kebutuhan fungsional dan non-fungsional.
    - Termasuk juga penyiapan data uji.
d. Test Environment Setup (Pengaturan Lingkungan Uji)
    - Mempersiapkan lingkungan yang diperlukan untuk menjalankan pengujian, seperti server, database, dan perangkat lunak pendukung.
e. Test Execution (Pelaksanaan Pengujian)
    - Menjalankan kasus uji dan mencatat hasilnya.
    - Bug atau error yang ditemukan akan dilaporkan menggunakan alat pelacak bug.
f. Test Cycle Closure (Penutupan Siklus Pengujian)
    - Mengevaluasi seluruh siklus pengujian, mendokumentasikan hasil akhir, pelajaran yang dipetik, dan menyampaikan laporan akhir pengujian.



Gambar 1. *Software Testing Life Cycle (STLC)*

Tahapan Pengujian *White Box testing*

Metodologi penelitian yang digunakan diambil dari *Software Testing Life Cycle (STLC)* pada langkah *ke 5 Test Execution* (Pelaksanaan Pengujian). Pada Test Execution (Pelaksanaan Pengujian) dilakukan tahapan Pengujian White Box testing yang berisikan antara lain: Analisa *flowchart/ Sourcode*, Flow Graph, Cylomatic Complexity, Jalur independen, Perancangan Test Case. Tahapan Pengujian tersebut seperti digambarkan pada gambar 2.

Gambar 2. Tahapan Pengujian White Box testing

Penjelasan dari gambar 2 adalah sebagai berikut :

- Analisa *flowchart/ Sourcode*, adalah representasi grafis dari suatu proses atau alur logika. Dalam pengujian perangkat lunak, flowchart digunakan untuk: memvisualisasikan proses bisnis atau logika sistem. berguna untuk memahami bagaimana alur program berjalan, menentukan titik kritis untuk pengujian (decision points, loops, *input-output*).
- Flow Graph adalah grafik terarah (*directed graph*) yang menggambarkan alur eksekusi program. Node (simpul) mewakili blok instruksi, dan edge (garis) menunjukkan arah eksekusi program.
- Cylomatic Complexity adalah ukuran kuantitatif dari jumlah jalur eksekusi independen dalam suatu program atau modul. Diperkenalkan oleh Thomas McCabe pada tahun 1976, metrik ini digunakan untuk: Menilai kompleksitas logika kode, Menentukan jumlah

minimum test case yang diperlukan untuk cakupan jalur penuh (*path coverage*). Membantu dalam proses *code refactoring* dan pemeliharaan kode.

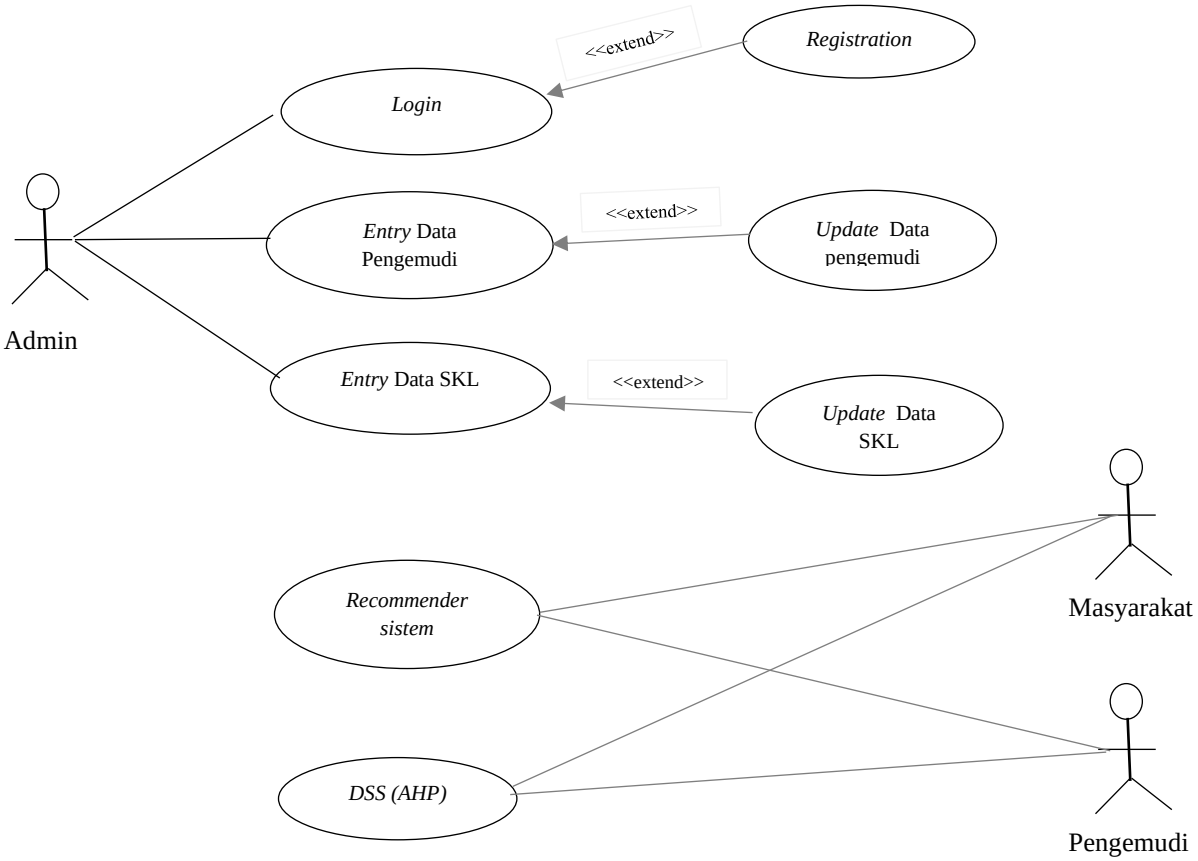
- Jalur independen adalah jalur dalam alur kontrol program (flow graph) yang melewati setidaknya satu edge baru (arah baru) yang belum dilalui oleh jalur lainnya.Mewakili kombinasi unik dari kondisi percabangan, loop, atau struktur logika lainnya. Tujuan adalah untuk: mengidentifikasi semua jalur logika utama dalam kode. Merancang test case minimal yang menyentuh seluruh logika program.

Perancangan Test Case

Setelah ditentukan jalur independen dari flow graph di atas, maka langkah selanjutnya adalah membuat test case minimal sejumlah jalur independen yang telah dibuat, memastikan bahwa setiap kemungkinan jalur yang akan dilewati dibuat test case-nya.

4. HASIL DAN PEMBAHASAN

Gambar 3 adalah *Use Case* Pengelola Pengemudi dan Standar Kompetensi Lulusan (SKL),



Gambar 3. *Use Case* Pengelola Pengemudi dan SKL

***Use Case Login* – Pengelola Pengemudi dan SKL**

Use case login pada pengelola pengemudi dan SKL adalah admin melakukan login ke menu utama. Untuk uraian deskripsi *use case* dijelaskan pada pada tabel 1

Tabel 1. Deskripsi Use Case Login- Pengelola Pengemudi dan SKL

Use Case Login - Pengelola pengemudi dan SKL	
Tujuan	Melakukan login
Deskripsi	Admin melakukan login, apabila belum *registrasi*, lakukan *registrasi*.
Aktor	Admin.
Skenario Utama	
Kondisi Awal	Aktor telah masuk *di website* pengemudi
Aksi Aktor	Reaksi Sistem
	Sistem merespon dengan menampilkan *form* Pengemudi form SKL, *recommender* sistem dan AHP
Kondisi Akhir	Aktor berhasil login, dan masuk ke sistem

Use Case Extend Registration

Use case Extend Registration pada pengelola pengemudi dan SKL adalah admin melakukan login ke menu utama. Apabila Admin belum melakukan *registrasi* maka lakukan r*egistrasi* terlebih dahulu. Untuk uraian deskripsi *use case* dijelaskan pada pada tabel 2

Tabel 2. Deskripsi *Use Case Extend Registration*

Use Case Extend Registration	
Tujuan	Melakukan *registrasi*
Deskripsi	Admin melakukan login, apabila belum *registrasi* lakukan *registrasi*.
Aktor	Admin.
Skenario Utama	
Kondisi Awal	Aktor telah masuk *di website* pengemudi
Aksi Aktor	Reaksi Sistem
	Sistem merespon dengan menampilkan *form registrasi*
Kondisi Akhir	Aktor telah melakukan *registrasi* sebagai admin

Use case Entry Data Pengemudi

Use case *entry* data pengemudi adalah admin melakukan memasukkan data pengemudi. Untuk uraian deskripsi *use case* dijelaskan pada pada tabel 3

Tabel 3. Deskripsi *Use Case Entry* Data Pengemudi

Use Case Entry Data Pengemudi	
Tujuan	Melakukan *entry* data pengemudi
Deskripsi	Admin melakukan login untuk memasukkan data pengemudi
Aktor	Admin.
Skenario Utama	
Kondisi Awal	Aktor telah masuk *di website* pengemudi
Aksi Aktor	Reaksi Sistem
	Sistem merespon dengan menampilkan *form* pengemudi
Kondisi Akhir	Aktor memasukkan data pengemudi

Use Case *Entry Data SKL*

Use Case Entry Data SKL adalah memasukkan data SKL. Untuk uraian deskripsi *use case* dijelaskan pada pada tabel 4

Tabel 4. Deskripsi *Use Case Entry* Data SKL

***Use Case Entry* Data SKL**	
Tujuan	Melakukan *Entry* Data SKL
Deskripsi	Admin melakukan login untuk memasukkan data SKL
Aktor	Admin.
Skenario Utama	
Kondisi Awal	Aktor telah masuk *di website* pengemudi
Aksi Aktor	Reaksi Sistem
	Sistem merespon menampilkan *form* SKL
Kondisi Akhir	Aktor memasukkan data SKL

Berikut dijelaskan menu pada sistem aplikasi Rekomendasi Penilaian Pengemudi Transportasi Umum Nasional, Menu terdiri dari Login, tampilan dasboard, profile pengemudi, Data Indikator Kelulusan, Sistem pertanyaan,Sistem Penilaian. Adapun menu menu dan tampilan menu dijelasan dibawah ini

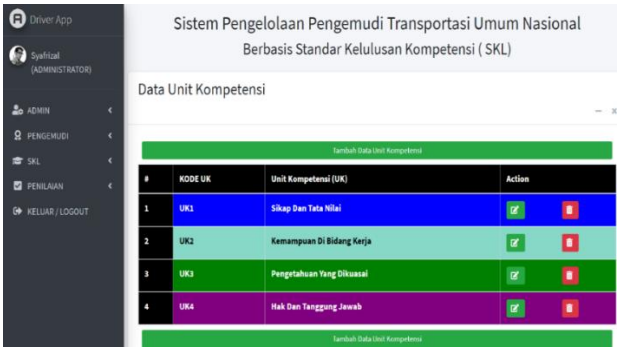
Tampilan Login

Tampilan Login Untuk Sistem Transportassi dapat dilihat pada gambar 1.

Gambar 4. Tampilan Login

Tampilan Dashboard

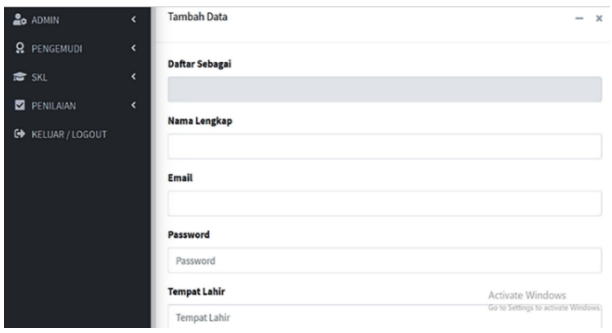
Menu utama terdiri dari administrator, pengemudi, penilaian, keluar / *logout* seperti pada gambar 6



Gambar 5. Menu Utama

***Profile* Pengemudi**

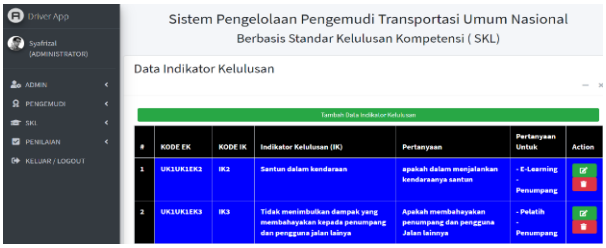
Gambar 7 menu utama, jika diklik pengemudi menampilkan *profile* pengemudi yang berisikan data pengemudi seperti nama, email, tanggal lahir, alamat.



Gambar 6. *Profile* Pengemudi

Data Indikator Kelulusan

Gambar 8 adalah menu menambah indikator kelulusan, jika ingin menambah indikator kelulusan dengan menekan tombol *action* lalu klik indikator kelulusan seperti pada gambar dibawah ini.



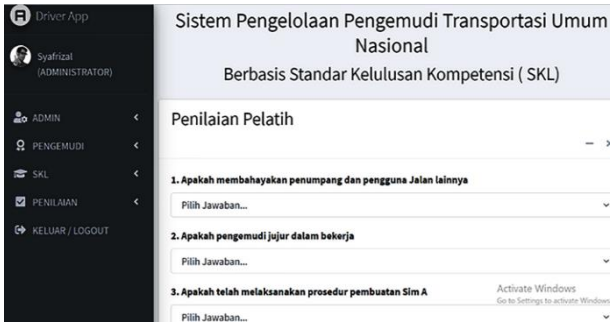
Gambar 7. Indikator Kelulusan

Sistem Penilaian

Sistem penilaian diambil dari data SKL pada 6 menu utama dengan memberi pertanyaan untuk *e-learning*, pengawas, pelatih, penumpang seperti pada gambar 9

Gambar 8. Sistem Penilaian

Dari gambar 9 yaitu misalkan pilih Pelatih, maka pertanyaan dilakukan oleh sistem , dan akan ditampilkan pertanyaan pertanyaan di menu pelatih.



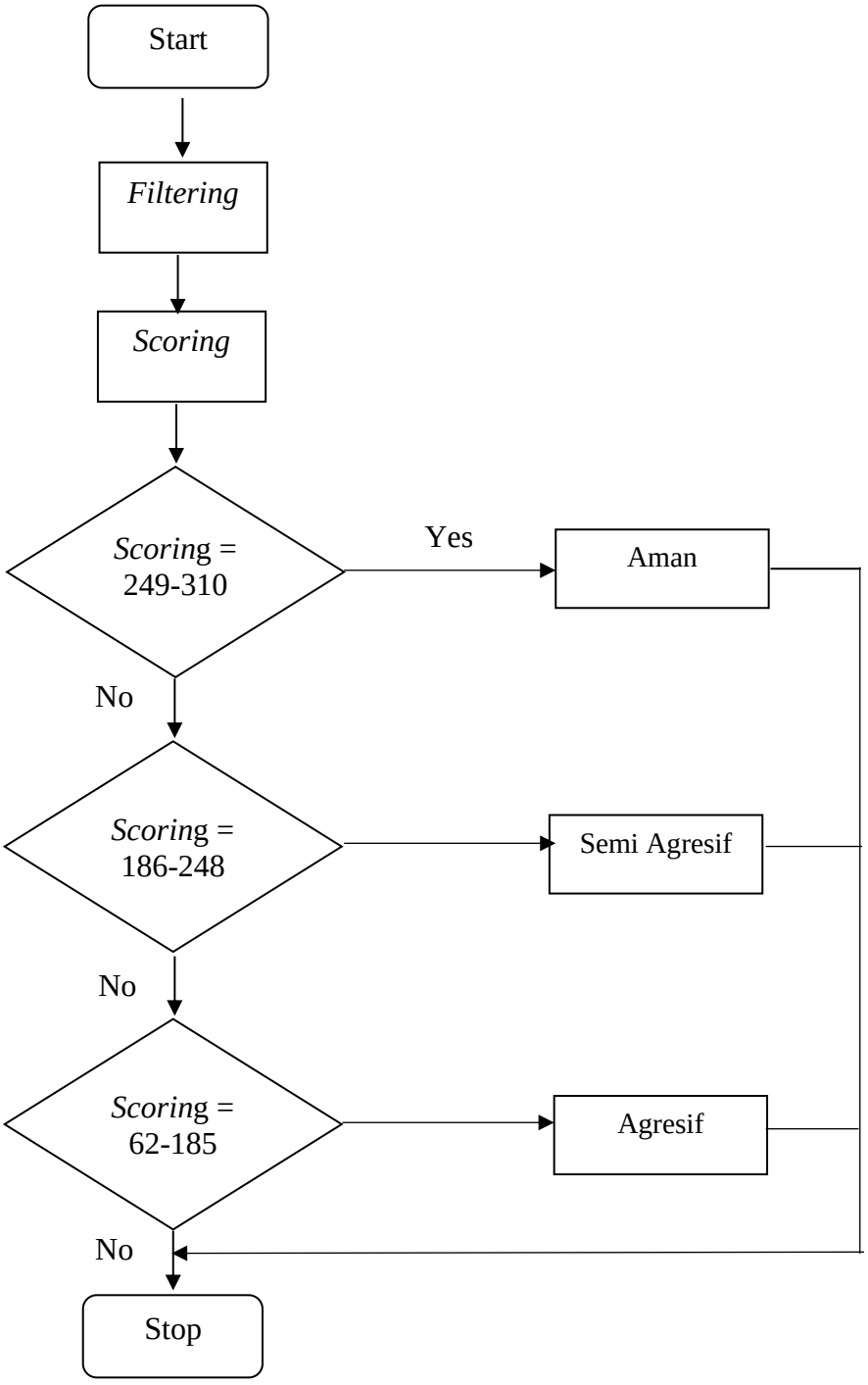
Gambar 9. Pertanyaan oleh Pelatih

Dari Aplikasi Pengemudi diatas diambil fitur tentang rekomendasi untuk menilai kompetensi dan perilaku pengemudi, untuk dilakukan pengujian white Box teting, yaitu Path

testing. Pada Fitur tersebut dilakukan tahapan tahapan pengujian yaitu analisa flowchart, flowgraph, Cylomatic Complexity, jalur independen dan Perancangan test case. Apun tahapan tahapan tersebut diuraikan scara rinci dibawah ini

Analisa *flowchart*

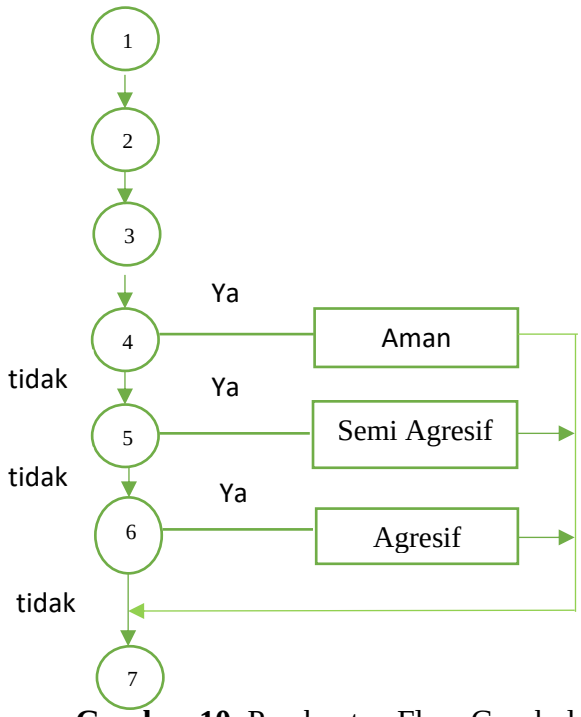
Dari usecase gambar 4 *Use Case* Pengelola Pengemudi dan SKL, dibuat flowchartnya sebagai berikut:



Gambar 10. *Use Case* Pengelola Pengemudi dan SKL

Flow Graph

Pembuatan Flow Graph dari *Flowchart* diatas adalah sebagai berikut:



Gambar 10. Pembuatan Flow Graph dari *Flowchart*

Cylomatic Complexity

Dilakukan perhitungan Cylomatic Complexity dari flowchart dari analisa flowchart pada tahap ke satu

V(G)= (E-N) +2

V(G)= Jumlah Region

E= Jumlah edge yang ditentukan dengan gambar panah

N= Jumlah simpul grafik (node) dengan gambar lingkaran

VG = (9- 7) + 2

VG = 4

Jalur independen

Dari Analisa Flowchart dilakukan penentuan jalur Independen. Jalur independen diperoleh sebagai berikut:

Jalur 1 =1,2,3,4, 7

Jalur 2 = 1,2,3,,5,7

Jalur 3 = 1,2,3, 6,7

Perancangan *Test Case*

Setelah ditentukan jalur independen dari flow graph di atas, maka langkah selanjutnya adalah membuat test case minimal sejumlah jalur independen yang telah dibuat, memastikan bahwa setiap kemungkinan jalur yang akan dilewati dibuat test case-nya.

Tabel 5. *Test Case*

Path	1
Jalur	1,2,3,4,7
Scenario	1. Start 2. Filtering 3. Scoring *4 Jika Scorin*g = 249-310, Aman 7. Stop
Hasil Pengujian	Berhasil
Path	2
Jalur	1,2,3,5,7
Scenario	1. Start 2. Filtering *1.* Scoring *5. Jika Scorin*g = 186-248 7. Stop
Hasil Pengujian	Berhasil
Path	3
Jalur	1,2,3, 6,7
Scenario	1. Start 2. Filtering 3. Scoring *6. Jika Scorin*g = 186-248 7. Stop
Hasil Pengujian	Berhasil

5. KESIMPULAN

Berdasarkan hasil pengujian sistem rekomendasi penilaian pengemudi transportasi umum nasional dengan metode *White Box Testing*, khususnya menggunakan pendekatan Path Testing, dapat disimpulkan bahwa: Struktur logika program telah berhasil diuji secara menyeluruh melalui identifikasi dan analisis jalur eksekusi (*path*) dalam kode sumber sistem. Kompleksitas siklomatik telah dihitung untuk menentukan jumlah jalur independen, dan seluruh jalur tersebut telah berhasil diuji tanpa adanya anomali atau kesalahan logika (*logic*

error). Tidak ditemukan cacat logika kritikal dalam proses pengambilan keputusan sistem, sehingga sistem dapat dikatakan stabil dan andal pada aspek struktur kendali programnya.

SARAN

Pemeliharaan kode secara berkala tetap diperlukan agar kompleksitas sistem tidak meningkat drastis seiring penambahan fitur di masa mendatang, yang dapat mengganggu keefektifan pengujian *white box*.Pertimbangkan untuk menerapkan automasi pengujian, terutama untuk pengujian jalur eksekusi berulang, guna meningkatkan efisiensi pengujian dan mengurangi kesalahan manusia.

UCAPAN TERIMA KASIH

Terima kasih penulis ucapkan kepada Asosiasi Pengemudi–Perkumpulan Bersama Pengerak Insan (PBPI), yang telah memfasilitasi dalam memperoleh data dan telah bersedia melakukan *Focus Group Discussion* (FGD). Terima kasih Kepada Departemen perhubungan Darat yang telah bersedia memberikan konfirmasi penelitian dan bersedia mekakukan *Focus Group Discussion.* Terima kasih kepada Lembaga Penelitian & Pengabdian Masyarakat Universitas Pembangunan Jaya yang telah mensupport untuk melakukan publikasi.

DAFTAR REFERENSI

Ali, A. H., & Saleem, N. N. (2023). Data Flow Testing and Tools Review. *Journal of Education and Science*, *32*(2), 51–60. https://doi.org/10.33899/edusj.2023.137611.1315

Andrian Ibrahim, R., & Saktian Laksito, G. (2024). Optimization of White Box Testing by Utilizing Branching and Repeating Structures in Java Programs Using Base Path. *International Journal of Mathematics, Statistics, and Computing*, *2*(2), 85–89.

Bardin, S., Chebaro, O., Delahaye, M., & Kosmatov, N. (2014). An all-in-one toolkit for automated white-box testing. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8570 LNCS*, 53–60. https://doi.org/10.1007/978-3-319-09099-3_4

Caniço, A., & Santos, A. (2023). Witter: A Library for White-Box Testing of Introductory Programming Algorithms. *SPLASH-E 2023 - Proceedings of the 2023 ACM SIGPLAN International Symposium on SPLASH-E, Co-Located with: SPLASH 2023*, 69–74. https://doi.org/10.1145/3622780.3623650

Chawan, P. M., Www, W. :, Thakare, S., Chavan, S., & Chawan, M. (2012). *Software Testing Strategies and Techniques International Journal of Emerging Technology and Advanced Engineering Software Testing Strategies and Techniques* (Vol. 2, Issue 4). www.ijetae.com

Dimas Saputro, A., & Azzahra Narwastika, A. (2023) *Implementasi White Box Testing Dengan Teknik Basis Path Pada Pengujian Form Peminjaman Sistem Aplikasi Perpustakaan*. Prosiding Seminar Nasional Teknologi Informasi dan Bisnis (SENATIB) 2023,e-ISSN 2962-1968

Golian, N., Golian, V., & Afanasieva, I. (2022). BLACK AND WHITE-BOX UNIT TESTING FOR WEB APPLICATIONS. *Bulletin of National Technical University "KhPI". Series: System Analysis, Control and Information Technologies*, *1 (7)*, 79–83. https://doi.org/10.20998/2079-0023.2022.01.13

Gusdevi, H., Kuswayati, S., Iqbal, M., Fikri, M., Bakar, A., Novianti, N., Ramadan, R., Studi, P., Informatika, T., Tinggi, S., & Bandung, T. (2022). *Pengujian White-Box Pada Aplikasi Debt Manager Berbasis Android*. *04*.

IEEE Standard for Software and System Test Documentation. (2008). IEEE. https://doi.org/10.1109/IEEESTD.2008.4578383

Izzat S, Saleem NN. (2023) Software Testing Techniques and Tools: A Review. Journal of Education and Science. Jun 1;32(2):31–40.

Kaner, C. (2001). *Measurement Issues & Software Testing Measurement Issues and Software Testing* www.kaner.com

Katlon. (2025.). Katlon,2025, White Box Testing: All You Need To Know, https://katalon.com/resources-center/blog/what-is-white-box-testing

Kumar, M., Professor, A., Kumar Singh, S., Dwivedi, R. K., & Professor, A. (2015). International Journal of Advance Research in Computer Science and Management Studies. *International Journal of Advance Research in Computer Science and Management Studies*, *3*(10). www.ijarcsms.com

Lammermann, F., & Wappler, S. (2018) *Benefits of Software Measures for Evolutionary White-Box Testing*. https://www.researchgate.net/publication/327691019

Li, N., Praphamontripong, U., & Offutt, J. (2009). An experimental comparison of four unit test criteria: Mutation, edge-pair, all-uses and prime path coverage. *IEEE International Conference on Software Testing, Verification, and Validation Workshops, ICSTW 2009*, 220–229. https://doi.org/10.1109/ICSTW.2009.30

Madhavi, D. (2016.). *A White Box Testing Technique in Software Testing: Basis Path Testing*. www.journalforresearch.or

Mega & Safrizal, 2025, Implementasi Pengujian White Box Menggunakan Path Testing untuk Meningkatkan Keandalan Sistem Presensi Berbasis Web di PT XYZ. In *JITUS: Journal Information Technology for Urban Society* (Vol. 1). Maret.https://ojs.upj.ac.id/index.php/JITUS

Nurwicaksono, M. A., Lisa, I. N., Tiara, A. R., & Sidik, R. (2023). Optimasi Sistem Informasi Konsultasi Hukum melalui Pendekatan Pengujian Kombinasi White-box dan Black-box. *Jurnal Manajemen Informatika (JAMIKA)*, *14*(1), 1–15. https://doi.org/10.34010/jamika.v14i1.10110

Peñalosa, Enrique., Votero, Fernando., Balencia, Paloma., & Barreras, Roi. (2019). *Licuadoras luminosas*. La Jaula Publicasiones.

Peraturan Pemerintah Republik Indonesia (2012),Peraturan Presiden Republik Indonesia Nomor 8 Tahun 2012 Tentang Kerangka Kualifikasi Nasional Indonesia, www.djpp.depkumham.go.id

Rahman Abdillah, Rudi Hermawan, Wawan Hermawansyah, Ibnu Adkha, & Heri Arifin. (2024). Pengujian Perangkat Lunak Sistem Informasi Inventori pada Usaha Jasa Pengiriman Paket. *Polygon : Jurnal Ilmu Komputer Dan Ilmu Pengetahuan Alam*, *2*(4), 166–175. https://doi.org/10.62383/polygon.v2i4.199

Rini, S. Y., & Kusmaya Putri, A. (2023.). Implementasi Blackbox Testing Dan Whitebox Testing Pada Pengujian Form Profil Toko Admin Sistem Aplikasi Raja Ongkir Berbasis Website. Prosiding Seminar Nasional Teknologi Informasi dan Bisnis (SENATIB),e-ISSN 2962-1968

Safrizal et al,2024,Testing Dan Implementasi, PT Mafy Media Literasi Indonesia , https://www.academia.edu/119966136/Buku_TESTING_DAN_IMPLEMENTASI_2024, 978-623-8638-29-1.

Sasmito, G. W. (2020a). White Box Testing with Basis Path Technique in the Demography Administration Website. *ICSECC 2020 - 2nd International Conference on Sustainable Engineering and Creative Computing, Proceedings*, 86–92. https://doi.org/10.1109/ICSECC51444.2020.9557428

Sasmito, G. W. (2020b). White Box Testing with Basis Path Technique in the Demography Administration Website. *ICSECC 2020 - 2nd International Conference on Sustainable Engineering and Creative Computing, Proceedings*, 86–92. https://doi.org/10.1109/ICSECC51444.2020.9557428

Sekhon, A., Ji, Y., Dwyer, M. B., & Qi, Y. (2022). *White-box Testing of NLP models with Mask Neuron Coverage*. http://arxiv.org/abs/2205.05050

Sheakh, T., Hussain, T., & Singh, S. (2015). International Journal of Allied Practice, Research and Review A Comparative Study of Software Testing Techniques Viz. White Box Testing Black Box Testing and Grey Box Testing. *IJAPRR International Peer Reviewed Refereed Journal*, *II*, 1–08. https://www.researchgate.net/publication/276028491

Suryanta, A., Wahyu Widianti, L., Mirsya Ashari, dan, Kodiklatad, P., & Jakarta STI, S. (2025). Pengujian White Box Terhadap Sistem Informasi Akademik (Siakad) Di Politeknik Angkatan Darat. In *Jurnal Mahasiswa Teknik Informatika)* (Vol. 9, Issue 2).

Syaikhuddin, M. M., Anam, C., Rinaldi, A. R., & Conoras, M. (2018). Conventional Software Testing Using White Box Method. Kinetik : Game Technology, Information System. *Computer Network, Computing, Electronics, and Control*, *3*(1), 67–74.

Žlahtič, B., Završnik, J., Blažun Vošner, H., & Kokol, P. (2024). Transferring Black-Box Decision Making to a White-Box Model. *Electronics (Switzerland)*, *13*(10). https://doi.org/10.3390/electronics13101895