TUGAS AKHIR

RANCANG BANGUN ALAT ANALISIS KERINGKASAN KODE SUMBER BERBASIS METRIK HALSTEAD PADA PROYEK JAVASCRIPT

Untuk Memenuhi Sebagian Persyaratan Mencapai Derajat Sarjana S-1 Program Studi Informatika

DISUSUN OLEH:

ZIDANE ZAHRAN LAZUAR PURNOMO

NIM. 22106050016

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA

YOGYAKARTA

2026

LEMBAR PENGESAHAN TUGAS AKHIR

KEMENTERIAN AGAMA
UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA
FAKULTAS SAINS DAN TEKNOLOGI
Jl. Marsda Adisucipto Telp. (0274) 540971 Fax. (0274) 519739 Yogyakarta 55281

PENGESAHAN TUGAS AKHIR

Nomor : B-1172/Un.02/DST/PP.00.9/06/2026

Tugas Akhir dengan judul : Rancang Bangun Alat Analisis Keringkasan Kode Sumber Berbasis Metrik Halstead Pada Proyek Javascript

yang dipersiapkan dan disusun oleh:

Nama : ZIDANE ZAHRAN LAZUAR PURNOMO
Nomor Induk Mahasiswa : 22106050016
Telah diujikan pada : Senin, 25 Mei 2026
Nilai ujian Tugas Akhir : A-

dinyatakan telah diterima oleh Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta

TIM UJIAN TUGAS AKHIR

Ketua Sidang

Dr. Fitri Wulandari, S.Si., M.Kom.
SIGNED

Valid ID: 6a1e733c224c0

Penguji I

Dr. Agung Fatwanto, S.Si., M.Kom.
SIGNED

Valid ID: 6a18070740e30

Penguji II

Muhammad Mustakim, S.T. M.T.
SIGNED

Valid ID: 6a1ab6b543db7

Yogyakarta, 25 Mei 2026
UIN Sunan Kalijaga
Dekan Fakultas Sains dan Teknologi

Prof. Dr. Dra. Hj. Khurul Wardati, M.Si.
SIGNED

Valid ID: 6a2130f728d09

LEMBAR KEASLIAN TUGAS AKHIR

SURAT PERNYATAAN KEASLIAN

Yang bertanda tangan dibawah ini:

Nama	: Zidane Zahran Lazuar Purnomo
NIM	: 22106050016
Program Studi	: Informatika
Fakultas	: Sains dan Teknologi

Menyatakan dengan sesungguhnya, bahwa tugas akhir saya yang berjudul **"Rancang Bangun Alat Analisis Keringkasan Kode Sumber Berbasis Metrik Halstead Pada Proyek Javascript"** merupakan hasil pekerjaan penulis sendiri sepanjang pengetahuan penulis, bukan duplikasi atau saduran dari karya orang lain kecuali bagian tertentu yang penulis ambil sebagai bahan acuan. Apabila terbukti pernyataan ini tidak benar, sepenuhnya menjadi tanggung jawab penulis.

Yogyakarta, 17 Mei 2026

Zidane Zahran Lazuar Purnomo
NIM. 22106050016

LEMBAR PERSETUJUAN TUGAS AKHIR

SURAT PERSETUJUAN TUGAS AKHIR

Hal : Persetujuan Tugas Akhir
Lamp : -

Kepada
Yth. Dekan Fakultas Sains dan Teknologi
UIN Sunan Kalijaga Yogyakarta
di Yogyakarta

Assalamu'alaikum wr. wb.

Setelah membaca, meneliti, memberikan petunjuk dan mengoreksi serta mengadakan perbaikan seperlunya, maka kami selaku pembimbing berpendapat bahwa skripsi Saudara:

Nama	:	Zidane Zahran Lazuar Purnomo
NIM	:	22106050016
Judul Tugas Akhir	:	Rancang Bangun Ekstensi Visual Studio Code dalam Mengukur Keringkasan Kode (*Conciseness*) Berbahasa JavaScript Menggunakan Metode Extreme Programming

sudah dapat diajukan kembali kepada Program Studi Informatika Fakultas Sains dan Teknologi UIN Sunan Kalijaga Yogyakarta sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata Satu dalam Program Studi Informatika.

Dengan ini kami mengharap agar skripsi/tugas akhir Saudara tersebut di atas dapat segera dimunaqasyahkan. Atas perhatiannya kami ucapkan terima kasih.

Wassalamu'alaikum wr. wb.

Yogyakarta, 17 Mei 2026
Pembimbing

Dr. Fitri Wulandari, S.Si., M.Kom.
NIP. 197410162000032002

LEMBAR PEDOMAN PENGGUNAAN TUGAS AKHIR

Tugas Akhir ini tidak dipublikasikan, tetapi tersedia di perpustakaan dalam lingkungan UIN Sunan Kalijaga, yang diperkenankan dipakai sebagai referensi kepustakaan, tetapi pengutipan harus seizin penyusun, dan harus menyebutkan sumbernya sesuai dengan kebiasaan ilmiah. Dokumen Tugas Akhir ini merupakan hak milik UIN Sunan Kalijaga Yogyakarta.

HALAMAN MOTTO

"*Everything not saved will be lost*"
-Nittendo *quit sccreen message*-

"*According to all aerodynamic laws the Bumblebee cannot fly because its body weight is not in the right proportion to its wingspan. But ignoring these laws, the Bee flies anyway.*"
-Peter Dinklage, Letters Live-

"*Thanks for the adventure, now go have a new one!*"
-Ellie's *Adventure Book,* UP (2009)-

HALAMAN PERSEMBAHAN

Dalam mengerjakan tugas akhir ini, hal utama bagi penulis adalah meniatkannya lillahi ta'ala. Dengan niat itu pula penulis dapat menyelesaikan tugas akhir ini dan mempersembahkannya khusus untuk:

Kedua Orang Tua Tercinta

Adik Saya

Almamater Tercinta

Program Studi Informatika

Fakultas Sains dan Teknologi

UIN Sunan Kalijaga Yogykarta

KATA PENGANTAR

Alhamdulillahi Rabbil 'Alamin, puji dan syukur penulis panjatkan ke hadirat Allah Subhanahu Wata'ala atas segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan tugas akhir yang berjudul "Rancang Bangun ekstensi Visual Studio Code dalam Mengukur Keringkasan Kode (*Conciseness*) Berbahasa Javascript Menggunakan Metode *Extreme Programming*" dengan baik. Shalawat serta salam senantiasa tercurah kepada junjungan kita, Nabi Muhammad Shallallahu 'Alaihi Wasallam, yang telah membawa umat manusia dari zaman kegelapan menuju zaman yang penuh dengan ilmu pengetahuan dan cahaya keimanan.

Penyusunan tugas akhir ini merupakan salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sunan Kalijaga Yogyakarta. Dalam proses penyusunannya, penulis menyadari bahwa keberhasilan penyelesaian karya ini tidak terlepas dari bantuan, bimbingan, serta dukungan dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Bapak Prof. Noorhaidi, S.Ag., M.A., M.Phil., Ph.D., selaku Rektor Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
2. Ibu Prof. Dr. Dra. Hj. Khurul Wardati, M.Si., selaku Dekan Fakultas Sains dan Teknologi, Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
3. Bapak Dr. Muhammad Mustakim, S.T., M.T., selaku Ketua Program Studi Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
4. Bapak Agus Mulyanto, S.Si., M.Kom., selaku Dosen Penasihat Akademik yang telah memberikan arahan dan bimbingan kepada penulis selama masa perkuliahan.
5. Ibu Dr. Fitri Wulandari, S.Si., M.Kom., selaku Dosen Pembimbing Tugas Akhir yang selalu meluangkan waktu, memberikan masukan berharga, serta dengan sabar membimbing penulis dalam setiap tahapan penyusunan tugas akhir ini.
6. Seluruh dosen dan karyawan Program Studi Informatika Universitas Islam Negeri Sunan Kalijaga Yogyakarta yang telah memberikan ilmu, pengalaman, serta bantuan selama masa studi penulis.
7. Kedua orang tua tercinta, Bapak Gandung Purnomo dan Ibu Eni Kisdaryati, atas segala doa, kasih sayang, serta dukungan yang tak pernah henti hingga penulis dapat menyelesaikan pendidikan tinggi ini.

8. Rekan-rekan seperjuangan Informatika angkatan 2022 yang telah menjadi bagian dari perjalanan akademik penulis, khususnya dalam memberikan semangat dan kebersamaan selama kuliah.
9. Rekan-rekan Gabut Bang yang tidak pernah berhenti menjadi teman belajar, bermain, maupun penulisan tugas akhir bagi penulis.
10. Kakak-kakak tingkat, Muhammad Arif Rakhman Azizi, Zildan Pandaru Sih Marginata, dan Maulana Yusuf Ahmadi yang selalu membantu dan memberikan masukan kepada penulis.
11. Rekan-rekan KKN Kelompok 28 Dusun Kleben yang telah memberikan motivasi kepada penulis, terutama dalam menjalani penyelesaian penulisan tugas akhir ini.
12. Terima kasih kepada teman-teman di luar lingkungan perkuliahan yang telah hadir sebagai tempat berbagi cerita, melepas penat, memberikan motivasi, dan membawa suasana yang lebih santai di tengah proses penyusunan tugas akhir ini.
13. Terakhir, terima kasih yang sebesar-besarnya kepada Sahabat Sapi yang telah menjadi sisi lain dari penulis sekaligus menjadi sosok yang paling mampu membuat penulis tetap merasa bahagia dan menikmati proses dalam perjalanan panjang penyusunan tugas akhir ini. Di tengah tekanan, kebingungan, dan rasa lelah yang datang silih berganti, kehadirannya menjadi pengingat bahwa proses ini tidak harus selalu dijalani dengan penuh beban. Sekali lagi, terima kasih karena telah menemani penulis untuk tetap berusaha, beristirahat sejenak dari kepenatan, dan terus bertahan hingga akhirnya tugas akhir ini dapat diselesaikan.

Penulis menyadari bahwa tugas akhir ini masih memiliki keterbatasan dan kekurangan, baik dari segi penyajian maupun substansi pembahasan. Oleh karena itu, penulis mengharapkan kritik dan saran yang bersifat membangun demi penyempurnaan karya ini di masa mendatang. Semoga tugas akhir ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan serta dapat dijadikan sebagai referensi bagi penelitian selanjutnya.

Yogyakarta, 15 Mei 2026
Penulis,

Zidane Zahran Lazuar Purnomo
NIM. 22106050016

INTISARI

Kualitas perangkat lunak merupakan aspek krusial dalam pengembangan sistem, di mana kualitas internal seperti keringkasan kode (*conciseness*) berpengaruh langsung terhadap biaya pemeliharaan dan keterbacaan program. Kode yang redundan dan bertele-tele meningkatkan beban kognitif pengembang dalam memahami logika sistem. Penelitian ini bertujuan untuk merancang dan membangun sebuah ekstensi pada *Integrated Development Environment* (IDE) Visual Studio Code yang mampu melakukan analisis statis untuk mengukur tingkat keringkasan kode JavaScript secara otomatis. Pengembangan sistem dilakukan menggunakan metode *Extreme Programming* (XP) melalui dua tahap iterasi yang mencakup perencanaan, perancangan, pengkodean, dan pengujian. Secara teknis, ekstensi ini dibangun menggunakan Node.js dengan memanfaatkan pustaka Acorn untuk mentransformasi kode sumber menjadi struktur *Abstract Syntax Tree* (AST). Pengukuran keringkasan kode didasarkan pada Model Kualitas McCall yang diimplementasikan melalui Metrik Halstead untuk kepadatan token. Hasil penelitian menunjukkan bahwa ekstensi *JS Conciseness Analyzer* berhasil menyajikan laporan kualitas kode secara *real-time* dengan tingkat akurasi perhitungan 99,99% berdasarkan validasi pengujian *blackbox*. Fitur tambahan berupa agregasi data *workspace* dan status kelayakan otomatis (*verdict*) membantu pengembang dalam menginterpretasikan hasil pengukuran dengan lebih mudah. Kesimpulannya, penelitian ini berhasil menyediakan alat bantu penganalisis statis yang efektif bagi pengembang JavaScript dalam memantau dan menjaga kualitas internal perangkat lunak mereka.

Kata Kunci: Keringkasan Kode, Metrik Halstead, *Cyclomatic Complexity*, Ekstensi Visual Studio Code, *Extreme Programming*, JavaScript.

ABSTRACT

Software quality is a crucial aspect of system development, where internal quality factors such as code conciseness directly impact maintenance costs and program readability. Redundant and verbose code increases the cognitive load on developers when understanding system logic. This study aims to design and develop a Visual Studio Code (VS Code) extension capable of performing static analysis to automatically measure JavaScript code conciseness. The system was developed using the Extreme Programming (XP) method through two iterations, including planning, design, coding, and testing. Technically, the extension is built on Node.js, utilizing the Acorn library to transform source code into an Abstract Syntax Tree (AST) structure. Code conciseness measurement is based on McCall's Quality Model, implemented via Halstead Metrics for token density and Cyclomatic Complexity for logical flow evaluation. The results indicate that the JS Conciseness Analyzer extension successfully provides real-time code quality reports with a 100% calculation accuracy rate, as verified by black-box testing. Additional features such as workspace data aggregation and automatic code status (verdict) assist developers in interpreting measurement results more easily. In conclusion, this research successfully provides an effective static analysis tool for JavaScript developers to monitor and maintain the internal quality of their software.

Keywords: *Code Conciseness, Halstead Metrics, Cyclomatic Complexity, Visual Studio Code Extension, Extreme Programming, JavaScript.*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1. Latar Belakang

Dalam rekayasa perangkat lunak, kualitas perangkat lunak tidak hanya dapat dinilai langsung oleh pengguna, seperti antarmuka dan pengalaman pengguna. Melainkan juga dengan kualitas internal kode sumber yang menyusunnya. Kualitas internal berperan penting dalam menentukan keberlanjutan suatu sistem perangkat lunak dalam jangka panjang. Studi evaluasi atribut kualitas perangkat lunak oleh Belinda et al. (2021) menyimpulkan bahwa *maintainability* menjadi aspek paling penting dan dominan sebesar 17,37%, disusul oleh *testability* sebesar 13,02%, dan *reliability* sebesar 10,35%. Perangkat lunak dengan *maintainability* yang rendah cenderung sulit dipelihara dan berisiko menimbulkan masalah *technical debt* (beban pengerjaan teknis) pada pengembangan fase selanjutnya [1]. Survei internasional InsighTD (653 responden industri IT) menemukan bahwa efek paling umum dari *technical debt* adalah *delivery delay* (keterlambatan pengiriman), *rework* (pengerjaan ulang), dan *increased effort* (peningkatan upaya/usaha) yang pada akhirnya akan mengarah kepada peningkatan biaya dan bahkan kerugian finansial[2].

Maintainability perangkat lunak tidak dapat diukur secara langsung karena bersifat abstrak. Oleh karena itu, diperlukan indikator kuantitatif yang dapat merepresentasikan kondisi internal kode sumber. Dalam model kualitas Perangkat Lunak McCall, salah satu indikator yang berkaitan dengan *maintainability* adalah keringkasan kode (*conciseness*). Keringkasan kode mengacu pada kemampuan perangkat lunak dalam memenuhi kebutuhan pengguna dengan kode seminimal mungkin [3]. Model kualitas McCall menekankan bahwa *maintainability* dipengaruhi oleh faktor-faktor seperti kemudahan kode dalam dipahami, diimplementasikan, ditelusuri, dan dimodifikasi di mana *conciseness* membantu mengurangi kompleksitas kode dan biaya perbaikan. Evaluasi kualitas sistem dengan McCall sering menyoroti kebutuhan peningkatan *maintainability* melalui pengurangan kode yang berlebihan dan perbaikan struktur kode agar lebih modular dan mudah dipahami. Dengan demikian, *conciseness* dalam konteks McCall bukan hanya soal jumlah baris kode, tetapi juga tentang efektivitas penyampaian fungsi tanpa duplikasi dan kerumitan yang tidak perlu[3].

Meskipun *conciseness* dinilai penting sebagai indikator *maintainability*, pengukurannya pada bahasa pemrograman yang dinamis dan fleksibel menghadirkan tantangan tersendiri seperti bahasa JavaScript [5]. Sebagai bahasa pemrograman yang paling banyak digunakan, fleksibilitas sintaksis JavaScript sering kali menyebabkan pengembang menuliskan kode redundan yang pada akhirnya akan menurunkan nilai *maintainability* perangkat lunak. Tanpa adanya alat bantu yang objektif, pengembang cenderung sulit menyadari adanya penumpukan kode yang tidak efektif selama fase pengembangan. Salah satu pendekatan ilmiah untuk mengukur *conciseness* kode sumber adalah melalui Metrik Halstead digunakan sebagai indikator utama untuk merepresentasikan *maintainability* melalui tingkat keringkasan kode [3]. Dalam penelitian ini, parameter skor *conciseness* dari Halstead berfokus pada penghitungan jumlah operator dan operan [5]. Selain itu, integrasi metrik *Cyclomatic Complexity* (CC) dan *Source Lines of Code* (SLOC) di tambahkan sebagai indikator pendukung untuk memberikan gambaran yang lebih jelas mengenai struktur logika dan baris kode efektif yang dihasilkan.

Berdasarkan permasalahan tersebut, diperlukan sebuah alat evaluasi yang tidak hanya akurat secara matematis tetapi juga praktis dalam siklus pengembangan perangkat lunak. Analisis program statis menjadi pendekatan pengukuran kualitas struktural pada level kode sumber (internal), karena mampu menilai kualitas struktur kode tanpa perlu mengeksekusi program tersebut. Hal ini juga didukung dengan kajian sistematis mengenai alat penilaian otomatis yang menunjukkan bahwa *tool* berbasis analisis statis sering kali digunakan untuk memeriksa *maintainability, readability*, dan dokumentasi melalui pemeriksaan gaya, pelanggaran *best practice*, dan keberadaan komentar [6]. Melalui pengembangan perkakas berbasis analisis statis yang terintegrasi langsung di dalam lingkungan *Integrated Development Environment* (IDE), pengembang dapat memantau tingkat kepadatan dan efisiensi semantik kode mereka secara otomatis, instan, dan dini, sebelum perangkat lunak tersebut memasuki tahap produksi.

1.2. Rumusan Masalah

Berdasarkan permasalahan tersebut, pertanyaan pengembangan yang diajukan dalam tugas akhir ini adalah “Bagaimana merancang dan mengimplementasikan sebuah ekstensi VS Code yang mampu melakukan analisis statis terhadap kode sumber untuk

menghasilkan skor keringkasan kode berdasarkan metrik Metrik Halstead, *Source Lines of Code* (SLOC*),* dan *Cyclomatic Complexity*(CC)?"

1.3. Batasan Masalah

Agar perancangan tugas akhir ini tetap terfokus dan sesuai dengan tujuan yang telah ditetapkan, maka batasan masalah dalam pengembangan ini adalah sebagai berikut:

a. Analisis statis dalam penelitian ini difokuskan pada evaluasi karakteristik internal kode sumber tanpa menilai perilaku program. Dengan demikian, sistem tidak ditujukan untuk mendeteksi kesalahan logika dan *bug runtime* program secara langsung, melainkan untuk memberikan gambaran kuantitatif mengenai struktur dan kompleksitas internal kode sumber.
b. Pengukuran keringkasan kode dilakukan dengan menggunakan 3 metrik yaitu Halstead, *Source Lines of Code* (SLOC), dan *Cyclomatic Complexity*,
c. Skor keringkasan yang dihasilkan bersifat indikatif dan digunakan sebagai alat bantu evaluasi, bukan sebagai standar baku penilaian *maintainability* perangkat lunak.
d. Ekstensi yang dikembangkan tidak melakukan perubahan atau *refactoring* kode sumber.

1.4. Tujuan Tugas Akhir

Penelitian ini bertujuan untuk merancang dan mengimplementasikan sebuah ekstensi yang mampu melakukan analisis statis terhadap kode sumber guna menghitung *conciseness* menggunakan Metrik Halstead serta *Source Lines of Code* (SLOC) dan *Cyclomatic Complexity* sebagai metrik dukungan. Selain itu, penelitian ini bertujuan untuk menghasilkan skor keringkasan kode yang bersifat indikatif dan dapat digunakan sebagai alat bantu evaluasi aspek *maintainability* kode sumber. Melalui integrasi langsung ke lingkungan pengembangan, ekstensi yang dikembangkan diharapkan dapat menyediakan umpan balik yang objektif dan konsisten secara langsung. Sehingga membantu developer dalam memantau kondisi keringkasan kode selama proses pengembangan perangkat lunak.

1.5. Manfaat Tugas Akhir

Penelitian dan pengembangan perangkat lunak ini diharapkan dapat memberikan manfaat yang nyata bagi berbagai pihak, baik secara praktis maupun teoretis di antaranya:

1.5.1 Bagi Pengembang Perangkat Lunak

1. Memberikan alat bantu praktis berupa ekstensi yang terintegrasi langsung dalam lingkungan pengembangan yaitu VSCode dan Antigravity. Sehingga pengembang dapat mengevaluasi tingkat keringkasan dan efisiensi kode JavaScript tanpa harus berpindah aplikasi.
2. Membantu pengembang dalam mengindikasikan redundansi atau ketidakefisienan logika program berskala besar (*workspace*) secara cepat.
3. Mendorong kebiasaan penulisan kode yang terstruktur dengan mempertimbangkan nilai ideal operator dan operan yang pada akhirnya akan menurunkan biaya dan waktu pemeliharaan perangkat lunak di masa depan.

1.5.2 Bagi Penulis

1. Menjadi sarana untuk mengimplementasikan teori-teori rekayasa perangkat lunak, khususnya indikator kualitas *maintainability* berdasarkan model kualitas McCall dan pengukuran kompleksitas kode ke dalam sebuah perangkat lunak yang fungsional.
2. Meningkatkan pemahaman dan keterampilan teknis secara mendalam terkait arsitektur pengembangan ekstensi.

1.5.3 Bagi Bidang Akademis dan Peneliti Selanjutnya

1. Memberikan bukti empiris dan studi kasus nyata mengenai relevansi penggunaan Metrik Halstead dalam mengukur sub-karakteristik *conciseness* pada bahasa pemrograman modern (JavaScript).
2. Menjadi rujukan arsitektural bagi peneliti lain yang ingin membangun instrumen *code analyzer* serupa.

BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil perancangan, implementasi, serta pengujian yang telah dilakukan terhadap ekstensi *JS Conciseness Analyzer*, maka dapat diambil beberapa kesimpulan sebagai berikut:

1. Penelitian ini telah berhasil membangun sebuah perangkat lunak pendukung pengembangan (*tooling*) berupa ekstensi Visual Studio Code yang mampu mengukur kualitas *miantainability* pada aspek keringkasan kode (*conciseness*) berdasarkan Model Kualitas McCall.
2. Penggunaan metode *Extreme Programming* (XP) dalam siklus pengembangan memungkinkan peneliti untuk melakukan iterasi perbaikan secara cepat, terutama dalam menanggapi umpan balik pengguna terkait standarisasi bahasa antarmuka dan akurasi klasifikasi token.
3. Hasil pengujian akurasi menunjukkan bahwa sistem memiliki tingkat presisi yang sangat tinggi sebesar 99,99% dalam melakukan analisis. Sedangkan hasil dari pengujian usabilitas menggunakan SUS, JS Conciseness Analyzer memperoleh skor 74,14 yang sudah tergolong ke dalam kategori *good* dan *acceptable.*

5.2. Saran

Meskipun sistem telah berfungsi dengan baik sesuai dengan tujuan penelitian, masih terdapat beberapa ruang pengembangan yang dapat dilakukan di masa mendatang terutama dalam beberapa saran dan masukan dalam pengembangan iterasi kedua, antara lain:

1. Pengembangan Antarmuka GUI: Mengembangkan antarmuka pengguna yang lebih interaktif menggunakan VS Code Webview API agar hasil analisis tidak hanya terbatas pada teks di *Output Panel*, melainkan dalam bentuk dasbor yang lebih modern.
2. Visualisasi Data: Menambahkan fitur visualisasi hasil analisis dalam bentuk grafik (seperti *bar chart* atau *radar chart*) untuk mempermudah pengembang dalam membandingkan tingkat kompleksitas dan kepadatan kode antar file di dalam satu proyek.

3. Sistem Rekomendasi Optimasi: Menambahkan fitur analisis cerdas yang mampu memberikan saran perbaikan atau rekomendasi optimasi secara otomatis pada bagian baris kode yang memiliki nilai kompleksitas tinggi atau skor kepadatan yang rendah.
4. Dukungan Bahasa Pemrograman Lain: Memperluas jangkauan penganalisis agar tidak hanya mendukung JavaScript, tetapi juga bahasa pemrograman lain yang populer di ekosistem VS Code seperti TypeScript atau Python.

DAFTAR PUSTAKA

[1] D. R. Wijendra, S. Lanka, dan K. P. Hewagamage, "Analysis of Cognitive Complexity with Cyclomatic Complexity Metric of Software General Terms," 2021.

[2] R. Ramač *dkk.*, "Prevalence, Common Causes and Effects of Technical Debt: Results from a Family of Surveys with the IT Industry," Sep 2021, doi: 10.1016/j.jss.2021.111114.

[3] Jim A. McCall, Paul K. Richards, dan Cene F. Walters, "Factors in Software Quality," *Final Technical Report from Rome Air Development Center*, vol. Volume 1 (of three), hlm. 1–168, Nov 1977.

[4] S. R. Wicaksono, R. Setiawan, dan M. Nurwegiono, "Jurnal Teknologi dan Manajemen Informatika Evaluasi Kualitas Sistem Informasi Pergudangan pada Perusahaan Distributor Obat-Obatan Menggunakan Model McCall," vol. 10, no. 2, hlm. 164–175, 2024, [Daring]. Tersedia pada: http://http://jurnal.unmer.ac.id/index.php/jtmi

[5] S. A. Abdulkareem dan A. J. Abboud, "Evaluating Python, C++, JavaScript and Java Programming Languages Based on Software Complexity Calculator (Halstead Metrics)," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1076, no. 1, hlm. 012046, Feb 2021, doi: 10.1088/1757-899x/1076/1/012046.

[6] A. Brasoveanu, M. Moodie, dan R. Agrawal, "Textual evidence for the perfunctoriness of independent medical reviews," dalam *CEUR Workshop Proceedings*, CEUR-WS, 2020, hlm. 1–9. doi: 10.1145/nnnnnnn.nnnnnnn.

[7] U. Iftikhar, N. Bin Ali, J. Börstler, dan M. Usman, "A tertiary study on links between source code metrics and external quality attributes," 1 Januari 2024, *Elsevier B.V.* doi: 10.1016/j.infsof.2023.107348.

[8] B. Curtis, R. A. Martin, dan P. E. Douziech, "Measuring the Structural Quality of Software Systems," *Computer (Long. Beach. Calif).*, vol. 55, no. 3, hlm. 87–90, Mar 2022, doi: 10.1109/MC.2022.3145265.

[9] M. Tito Ramadhan, Y. Gunawan, K. Manaf, H. Purwanto, R. S. Perdana, dan B. Subaeki, "Testing the Autxhentication Platform System Using the Mccall Model," *CoreID Journal*, vol. 2, no. 1, hlm. 20–26, Mar 2024, doi: 10.60005/coreid.v2i1.26.

[10] Ahmad Farisi, D. Dafid, dan Rizani Teguh, "Analisis Metode Pengukuran Kualitas Perangkat Lunak: Sebuah Tinjauan Literatur Sistematis," *SATESI: Jurnal Sains Teknologi dan Sistem Informasi*, vol. 4, no. 1, hlm. 10–16, Apr 2024, doi: 10.54259/satesi.v4i1.2551.

[11] B. Khan dan A. Nadeem, "Evaluating the effectiveness of decomposed Halstead Metrics in software fault prediction," *PeerJ Comput. Sci.*, vol. 9, 2023, doi: 10.7717/peerj-cs.1647.

[12] V. K. L. Srivastava, N. Chandra Sekhar Reddy, dan A. Shrivastava, "An efficient software source code metrics for implementing for software quality analysis," *International Journal of Emerging Trends in Engineering Research*, vol. 7, no. 9, hlm. 216–222, Sep 2019, doi: 10.30534/ijeter/2019/01792019.

[13] D. Landman, A. Serebrenik, E. Bouwers, dan J. J. Vinju, "Empirical analysis of the relationship between CC and SLOC in a large corpus of Java methods and C functions," *Journal of Software: Evolution and Process*, vol. 28, no. 7, hlm. 589–618, Jul 2016, doi: 10.1002/smr.1760.

[14] L. Ardito, R. Coppola, L. Barbato, dan D. Verga, "A Tool-Based Perspective on Software Code Maintainability Metrics: A Systematic Literature Review," 2020, *Hindawi Limited*. doi: 10.1155/2020/8840389.

[15] T. Brito *dkk.*, "Study of JavaScript Static Analysis Tools for Vulnerability Detection in Node.js Packages," Jan 2023, doi: 10.1109/TR.2023.3286301.

[16] A. Shukla, "Modern JavaScript Frameworks and JavaScripts Future as a Full- Stack Programming Language," *Journal of Artificial Intelligence & Cloud Computing*, vol. 2, no. 4, hlm. 1, Okt 2023, doi: 10.47363/JAICC/2023(2)144.

[17] S. Mujahid, R. Abdalkareem, dan E. Shihab, "What are the characteristics of highly-selected packages? A case study on the npm ecosystem," Apr 2022, [Daring]. Tersedia pada: http://arxiv.org/abs/2204.04562

[18] Y. Wang dan H. Li, "Code Completion by Modeling Flattened Abstract Syntax Trees as Graphs," 2021. [Daring]. Tersedia pada: www.aaai.org

[19] X. Jiang, Z. Zheng, C. Lyu, L. Li, dan L. Lyu, "TreeBERT: A Tree-Based Pre-Trained Model for Programming Language," Jul 2021, [Daring]. Tersedia pada: http://arxiv.org/abs/2105.12485

[20] M. Lamada, A. Bakry, A. Z. Ifani, dan K. Khaerunnisa, "Development of Web-Based Project Tender Documents Application Using Extreme Programming Methods," *Elinvo (Electronics, Informatics, and Vocational Education)*, vol. 7, no. 2, hlm. 101–111, Feb 2023, doi: 10.21831/elinvo.v7i2.49863.

[21] S. A. Abdulkareem dan A. J. Abboud, "Evaluating Python, C++, JavaScript and Java Programming Languages Based on Software Complexity Calculator (Halstead Metrics)," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 1076, no. 1, hlm. 012046, Feb 2021, doi: 10.1088/1757-899x/1076/1/012046.

[22] R. Zaidi, "Operators in JavaScript," dalam *JavaScript Essentials for SAP ABAP Developers*, Apress, 2017, hlm. 31–47. doi: 10.1007/978-1-4842-2220-1_3.

[23] G. B. Balogun, M. Abdulraheem, P. O. Sadiku, O. D. Taofeek, dan A. Adewale, "Halstead's Complexity Measure of a Merge sort and Modified Merge sort Algorithms," *Jambura Journal of Informatics*, vol. 5, no. 2, hlm. 141–151, Nov 2023, doi: 10.37905/jji.v5i2.21995.

[24] P. Wang, C. Brown, J. A. Jennings, dan K. T. Stolee, "Demystifying Regular Expression Bugs: A comprehensive study on regular expression bug causes, fixes, and testing," Apr 2021, [Daring]. Tersedia pada: http://arxiv.org/abs/2104.09693

[25] K. Orfanou, N. Tselios, dan C. Katsanos, "Perceived Usability Evaluation of Learning Management Systems: Empirical Evaluation of the System Usability Scale."

[26] J. Brooke, "SUS - A quick and dirty usability scale."

[27] D. M. D. U. Putra, A. S. Kusuma, A. G. Willdahlia, dan N. K. N. N. Pande, "Evaluasi Usability E-Modul Basis Data Menggunakan Metode System Usability Scale (SUS),"

Jurnal Ilmiah Global Education, vol. 5, no. 2, hlm. 1800–1809, Jun 2024, doi: 10.55681/jige.v5i2.2764.

[28] A. Bangor, P. Kortum, dan J. Miller, “Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale,” 2009.