TUGAS AKHIR

RANCANG BANGUN EXTENSION VSCODE UNTUK CODE REVIEW OTOMATIS BERBASIS AI GENERATIF

Sebagai Memenuhi Persyaratan Mencapai Derajat Sarjana (S1)

DISUSUN OLEH:

HIRZIL KAISAN

NIM.21106050078

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA

YOGYAKARTA

2026

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-660/Un.02/DST/PP.00.9/04/2026

Tugas Akhir dengan judul : RANCANG BANGUN EXTENSION VSCODE UNTUK CODE REVIEW OTOMATIS BERBASIS AI GENERATIF

yang dipersiapkan dan disusun oleh:

Nama : HIRZIL KAISAN
Nomor Induk Mahasiswa : 21106050078
Telah diujikan pada : Kamis, 12 Maret 2026
Nilai ujian Tugas Akhir : A/B

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

TIM UJIAN TUGAS AKHIR

Ketua Sidang

Dr. Ir. Aulia Faqih Rifa'i, M.Kom.
SIGNED

Valid ID: 69deb8164302e

Penguji I

Ir. Muhammad Didik Rohmad Wahyudi, S.T., MT.
SIGNED

Valid ID: 69df37e06f0e8

Penguji II

Eko Hadi Gunawan, M.Eng.
SIGNED

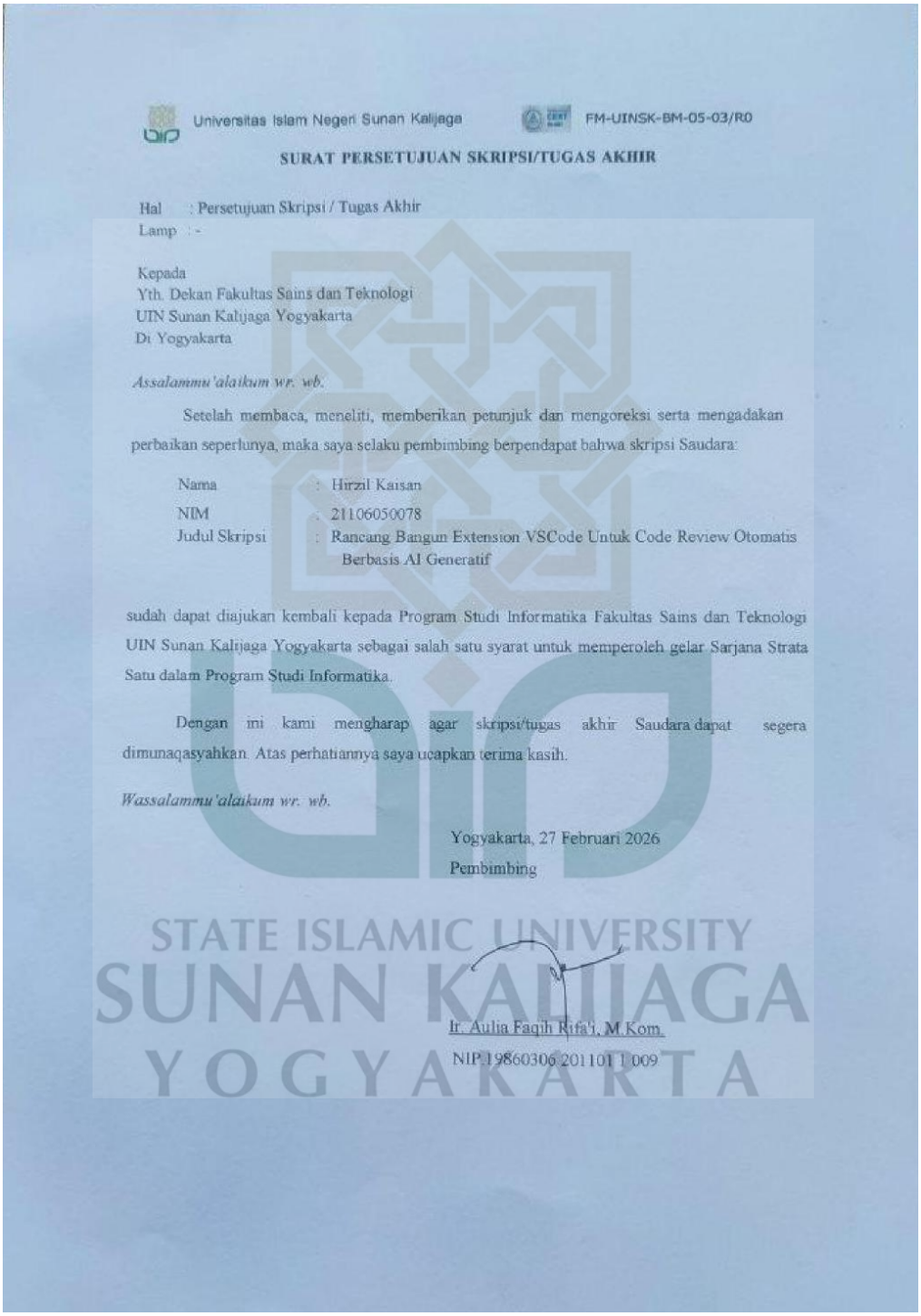
Valid ID: 69ddb7aef2484

Yogyakarta, 12 Maret 2026
UIN Sunan Kalijaga
Dekan Fakultas Sains dan Teknologi

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

Valid ID: 69df3d12a66f8

SURAT PERSETUJUAN TUGAS AKHIR



Universitas Islam Negeri Sunan Kalijaga FM-UINSK-BM-05-03/R0

SURAT PERSETUJUAN SKRIPSI/TUGAS AKHIR

Hal : Persetujuan Skripsi / Tugas Akhir
Lamp : -

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

Assalammu'alaikum wr. wb.

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

Nama	:	Hirzil Kaisan
NIM	:	21106050078
Judul Skripsi	:	Rancang Bangun Extension VSCode Untuk Code Review Otomatis Berbasis AI Generatif

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 dapat segera dimunaqasyahkan. Atas perhatiannya saya ucapkan terima kasih.

Wassalammu'alaikum wr. wb.

Yogyakarta, 27 Februari 2026
Pembimbing

Ir. Aulia Faqih Rifa'i, M.Kom.
NIP.19860306 201101 1 009

SURAT PERNYATAAN KEASLIAN TUGAS AKHIR

SURAT PERNYATAAN KEASLIAN/BEBAS PLAGIASI

Saya yang bertanda tangan dibawah ini:

Nama	: Hirzil Kaisan
NIM	: 21106050078
Program Studi	: Informatika
Fakultas	: Sains dan Teknologi

Dengan ini menyatakan bahwa isi tugas akhir saya yang berjudul "RANCANG BANGUN EXTENSION VSCODE UNTUK CODE REVIEW OTOMATIS BERBASIS AI GENERATIF" merupakan hasil pemikiran penulis sendiri, tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan orang lain sepengetahuan penulis, kecuali yang tertulis dalam tugas akhir dan disebutkan dalam daftar pustaka.

Demikian surat pernyataan ini saya buat untuk dipergunakan sebagaimana mestinya.

Yogyakarta, 27 Februari 2026

Hirzil Kaisan

NIM.21106050078

ABSTRAK

Perkembangan teknologi perangkat lunak menuntut programmer, terutama pemula, untuk menulis kode yang mudah dibaca, efisien, dan minim bug. Studi sebelumnya menunjukkan bahwa program buatan pemula kerap mengandung pelanggaran gaya penulisan, dokumentasi buruk, serta pola kode yang membingungkan (*atoms of confusion*), yang menurunkan *maintainability*.

Penelitian ini mengembangkan ekstensi Visual Studio Code berbasis AI generatif untuk *code review* otomatis, dengan fitur pemilihan model AI, konsep review seperti *clean code*, serta cakupan analisis kode. Hasil review disajikan melalui Webview interaktif dengan opsi penyimpanan saran perbaikan.

Integrasi GROQ API dan antarmuka interaktif diharapkan membantu programmer pemula memahami kualitas kode mereka, mendeteksi serta memperbaiki pelanggaran, dan meningkatkan produktivitas. Berdasarkan literatur, kode berkualitas rendah dapat mengandung hingga 15 kali lebih banyak bug dibandingkan kode bersih, sehingga tool ini berpotensi signifikan dalam mengurangi bug dan mempercepat proses perbaikan.

Penelitian ini berkontribusi secara praktis dalam mendukung penulisan kode berkualitas bagi pemula, dan secara akademis memberikan *baseline* implementasi *AI-driven code review* sebagai pendekatan baru di bidang pendidikan dan industri pemrograman. Selain itu, extension dilengkapi dengan fitur interactive chat yang memungkinkan pengguna bertanya langsung kepada AI untuk memperoleh klarifikasi atau penjelasan lebih lanjut terhadap hasil *review*.

Kata kunci: *VSCode extension*; AI generatif; *Code review* otomatis; *Code quality*; Pemula pemrograman; *Code inefficiency*; *Clean code*; *Developer productivity*.

ABSTRACT

The rapid advancement of software technology requires programmers, especially beginners, to write code that is readable, efficient, and low in bugs. Previous studies indicate that novice programmers often produce code containing style violations, poor documentation, and confusing code patterns (atoms of confusion), which reduce maintainability.

This research designs and develops a Visual Studio Code extension powered by generative AI for automated code review. The extension enables users to select AI models and review concepts (e.g., clean code principles) as well as determine whether to review the entire code or specific lines. The review results are presented through an interactive Webview, with an option to save suggested code improvements.

With the integration of the GROQ API and an interactive interface, this extension is expected to assist novice programmers in understanding their code quality, detecting and fixing code violations, and improving coding efficiency and productivity. Literature suggests that low-quality code may contain up to fifteen times more bugs than clean code, highlighting the potential of this tool to reduce software defects and accelerate the review and debugging process.

This study contributes practically by supporting beginners in writing high-quality code and academically by providing a baseline for implementing AI-driven code review as a novel approach in programming education and software industry practice. Additionally, the extension features an interactive chat that allows users to ask follow-up questions to the AI for further clarification or explanation of the review results.

Keywords: *VSCode extension*; *Generative AI; Automated code review; Code quality; Beginner programmers; Clean code; Developer productivity.*

KATA PENGANTAR

Puja dan puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir dengan judul "Rancang Bangun *Extension VSCode* untuk *Code Review* Otomatis Berbasis AI Generatif." Sholawat serta salam senantiasa penulis curahkan kepada yang terhormat baginda Nabi Muhammad SAW, yang telah memberikan syafaat serta telah menjadi teladan yang luar biasa bagi umat islam dalam menuntut ilmu serta mengamalkannya.

Terselesaikannya tugas akhir ini tidak luput dari bantuan serta dukungan berbagai pihak yang senantiasa menemani, memberikan semangat, bimbingan, dan juga ikut serta dalam berkontribusi selama proses penyusunan. Oleh karena itu, dengan kerendahan hati dan penuh rasa syukur, penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Bapak Prof. Noorhaidi Hasan, 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 Muhammad Mustakim, S.T., M.T., selaku Ketua Program Studi Informatika Fakultas Sains dan Teknologi Universitas Islam Negeri Sunan Kalijaga Yogyakarta.
4. Bapak Ir. Aulia Faqih Rifa'i, M.Kom., selaku Dosen Pembimbing Tugas Akhir yang telah meluangkan waktu untuk membantu dan mengarahkan selama proses penulisan tugas akhir.
5. Bapak Eko Hadi Gunawan, M.ENG., selaku Dosen Pembimbing Akademik yang telah membantu penulis selama perkuliahan.
6. Seluruh Dosen dan Karyawan Program Studi Informatika UIN Sunan Kalijaga yang telah memberikan ilmu dan bantuan selama perkuliahan.
7. Kedua orang tua penulis yaitu Bapak Ajimuddin, S.Ag., dan Ibu Siti Fajriyah yang senantiasa memberikan bantuan dan dukungan kepada

penulis sehingga dapat menyelesaikan studinya di perguruan tinggi UIN Sunan Kalijaga yang sangat dibanggakan ini.

8. Fansuri Fadel, Aenur Rokhman, Hanif Fajar A., Rizal Darusman, Indra Arya, Ihfansyah Maulana, Rina Agustina, Madinatuzzahro, serta seluruh rekan Informatika yang turut membantu penulis selama perkuliahan terutama teman-teman Informatika 2021 yang telah menjadi saksi perjuangan penulis pada saat perkuliahan.

Tugas akhir ini tidaklah sempurna dan tentu masih perlu beberapa berbaikan. Untuk itu penulis memohon kritik dan saran yang membangun agar dapat membawa manfaat bagi para pembaca nantinya. Semoga tugas akhir ini dapat memberikan pengetahuan yang baru dan dapat dijadikan referensi pembaca untuk kedepannya.

Yogyakarta, 26 Februari 2026

Penulis

Hirzil Kaisan

NIM.21106050078

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi perangkat lunak menuntut programmer pemula untuk menulis kode yang tidak hanya berfungsi, tetapi juga mudah dibaca, efisien, dan minim bug. Penelitian Kohlbacher dkk. (2023) terhadap lebih dari 13.000 file kode mahasiswa menemukan bahwa pelanggaran kualitas seperti kesalahan gaya penulisan, redundansi, dokumentasi lemah, serta pola *atoms of confusion* merupakan masalah dominan yang menurunkan *maintainability* dan meningkatkan risiko bug di tahap selanjutnya [1]. Konsep *atoms of confusion* sendiri pertama kali diidentifikasi oleh Gopstein dkk. (2017) sebagai struktur kode yang secara sintaks benar namun membingungkan secara logika, yang terbukti meningkatkan kesalahan pembacaan [11]. Tantangan belajar pemrograman pada pemula, seperti kesulitan memahami abstraksi dan struktur kode, juga telah menjadi perhatian dalam pendidikan ilmu komputer [20].

Studi Langhout & Aniche (2021) memperkuat temuan ini, menunjukkan bahwa programmer pemula 2,7 hingga 56 kali lebih rentan berbuat salah saat membaca atau memodifikasi kode yang melanggar prinsip penulisan yang baik [2]. Dampaknya tidak hanya menurunkan kualitas pembelajaran, tetapi juga menghambat produktivitas dan pemahaman dalam proses belajar mandiri mahasiswa atau developer junior.

Dari perspektif industri, laporan Thornhill & Borg (2022) mengungkap bahwa kode berkualitas rendah mengandung 15 kali lebih banyak bug dan memerlukan waktu *troubleshooting* hingga 124% lebih lama [3]. Temuan ini menegaskan bahwa kualitas kode merupakan faktor krusial bagi efisiensi tim dan keberlanjutan proyek perangkat lunak, sehingga pemahaman dini terhadap standar kode harus didukung tools yang tepat.

Perkembangan AI generatif membuka peluang baru untuk mengatasi gap ini. Model AI modern mampu memahami konteks kode, mendeteksi pelanggaran, memberikan saran perbaikan, dan menjelaskan logika dalam bahasa sederhana. *Visual Studio Code (VSCode)* sebagai editor terpopuler mendukung pengembangan *extension custom*, dan sudah ada beberapa *extension* terintegrasi AI, namun belum ada yang secara spesifik mengintegrasikan AI generatif untuk membantu khusus *code review* dan secara fleksibel dapat memilih model AI yang ingin digunakan (Llama, gpt, dll), dan dengan pemilihan konsep review spesifik (*clean code*, *security, performance, bug detection, documentation*), serta opsi tinjauan parsial kode dengan penyimpanan rekomendasi dan juga fitur *interactive chat* dengan beberapa opsi model AI yang disediakan.

Berdasarkan permasalahan tersebut, penelitian ini merancang dan membangun *extension* VSCode berbasis GROQ API untuk *code review* otomatis. Fitur utama meliputi pemilihan model AI, konsep review, cakupan kode (seluruh file atau baris tertentu), tampilan hasil interaktif via Webview, serta *interactive chat*. Solusi ini diharapkan membantu programmer pemula memahami kualitas kode, meminimalkan kesalahan, dan meningkatkan efisiensi pengembangan.

1.2 Rumusan Masalah

Bagaimana merancang dan mengimplementasikan *extension Visual Studio Code* berbasis AI generatif yang terintegrasi dengan GROQ API untuk melakukan *code review* otomatis secara fleksibel dan dengan fitur pemilihan model AI, pemilihan konsep review, kemampuan meninjau sebagian maupun seluruh kode, serta *interactive chat* sehingga nantinya dapat membantu meningkatkan kualitas dan pemahaman kode pada programmer pemula?

1.3 Tujuan

Adapun tujuan yang diharapkan penulis pada akhir penelitian ini adalah untuk merancang dan membangun *extension Visual Studio Code* berbasis AI generatif yang terintegrasi dengan GROQ API guna mendukung proses *code review* otomatis secara fleksibel, interaktif, dan mudah digunakan, dengan menyediakan fitur pemilihan model AI, pemilihan konsep atau metode review, kemampuan

meninjau sebagian maupun keseluruhan kode secara efisien, serta *interactive chat* sehingga pengguna dapat bertanya lebih lanjut tentang hasil *review* yang ditampilkan.

1.4 Manfaat

Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan pemanfaatan AI generatif di bidang rekayasa perangkat lunak, khususnya pada implementasi *code review* otomatis dalam lingkungan pengembangan terintegrasi (IDE). Selain itu, penelitian ini dapat menjadi referensi bagi penelitian selanjutnya yang berkaitan dengan integrasi model AI dalam proses peningkatan kualitas kode dan pembelajaran pemrograman. Diharapkan juga penelitian ini dapat memberikan manfaat bagi berbagai pihak terkait :

1. **Bagi Programmer Pemula**
   Extension yang dikembangkan diharapkan dapat membantu programmer pemula memahami kualitas kode yang ditulis, mendeteksi kesalahan, serta memperoleh saran perbaikan secara langsung sehingga meningkatkan pemahaman terhadap prinsip penulisan kode yang baik.
2. **Bagi Dunia Akademik**
   Penelitian ini dapat menjadi alternatif solusi dalam mendukung proses pembelajaran pemrograman, terutama dalam membantu mahasiswa memperoleh umpan balik otomatis tanpa ketergantungan penuh pada *review* manual dari dosen atau mentor.
3. **Bagi Industri Perangkat Lunak**
   Hasil penelitian ini diharapkan dapat menjadi solusi pendukung dalam proses *code review* yang lebih cepat dan efisien, sehingga membantu meningkatkan produktivitas serta mengurangi potensi bug akibat kualitas kode yang kurang baik.

1.5 Batasan Masalah

Agar penelitian ini lebih terarah dan tidak meluas dari tujuan yang telah ditetapkan, maka batasan masalah dalam penelitian ini adalah sebagai berikut :

1. Sistem yang dikembangkan berupa *extension* pada *Visual Studio Code (VSCode)* dan tidak mencakup pengembangan IDE atau editor lain.
2. Integrasi AI generatif dilakukan melalui GROQ API, sehingga performa dan hasil review bergantung pada model AI yang tersedia pada platform tersebut.
3. Fitur *code review* yang dikembangkan terbatas pada analisis kualitas kode berdasarkan konsep tertentu seperti *clean code, security, performance, bug detection*, dan *documentation*, sesuai dengan opsi yang disediakan dalam sistem.
4. *Extension* memungkinkan peninjauan kode secara keseluruhan maupun sebagian (*selected lines*), namun tidak mencakup analisis berbasis repository penuh atau multi-file dependency secara kompleks.
5. Sistem tidak menggantikan peran *code review* manual secara menyeluruh, melainkan berfungsi sebagai alat bantu pendukung dalam meningkatkan kualitas dan pemahaman kode.
6. Pengujian dan evaluasi sistem difokuskan pada efektivitas penggunaan oleh programmer pemula, dan tidak mencakup pengujian skala industri besar atau enterprise-level deployment.
7. Penelitian ini tidak membahas pelatihan (training) atau pengembangan model AI baru, melainkan memanfaatkan model AI generatif yang telah tersedia melalui API.
8. Kualitas dari jawaban *interactive chat* tergantung kepada model AI yang dipilih.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang telah dilakukan pada penelitian ini, sistem berhasil mengintegrasikan teknologi *Large Language Model* berbasis arsitektur transformer ke dalam lingkungan pengembangan Visual Studio Code melalui Groq API. Hal ini menunjukkan bahwa pendekatan AI generatif dapat diimplementasikan secara praktis sebagai alat bantu peningkatan kualitas kode bagi programmer pemula.

Adapun kesimpulan yang dapat diambil dari penelitian ini adalah sebagai berikut:

1. Penelitian ini berhasil merancang dan membangun sebuah *extension Visual Studio Code* berbasis AI generatif yang dapat digunakan untuk melakukan *code review* otomatis dengan memanfaatkan GROQ API.
2. *Extension* yang dikembangkan mampu menjalankan fungsi utama berupa melakukan analisis kode, menampilkan hasil *review*, serta memberikan rekomendasi perbaikan berdasarkan metode *review* yang dipilih, seperti *Clean Code, Security, Performance, Bug Detection*, dan *Documentation*.
3. Sistem menyediakan fitur pemilihan model AI sehingga pengguna dapat menyesuaikan kebutuhan antara kecepatan respons dan kedalaman analisis, yang terbukti dari perbedaan waktu respons pada masing-masing model yang diuji.
4. Seluruh fitur yang dirancang berjalan dengan lancar, meliputi *review* kode secara keseluruhan maupun parsial (*selected lines*), pemilihan metode *review*, serta penyimpanan kode hasil rekomendasi (*suggested code*). Selain itu, sistem juga dilengkapi dengan fitur *interactive chat* yang memungkinkan pengguna, khususnya programmer pemula, untuk melakukan tanya jawab lanjutan guna memperoleh klarifikasi lebih mendalam terhadap hasil *review*.
5. Secara keseluruhan, *extension Code Review Assistant* dapat membantu programmer, khususnya pemula, dalam memahami kualitas kode,

mempercepat proses *code review*, serta meningkatkan interaksi edukatif melalui fitur chat, sehingga mendukung peningkatan efisiensi dan kualitas pengembangan perangkat lunak.

5.2 Saran

Pengembangan lebih lanjut dapat mempertimbangkan optimalisasi prompt engineering [18][19] serta eksplorasi model LLM terbaru yang memiliki kemampuan reasoning lebih baik sebagaimana ditunjukkan pada penelitian GPT-4 [9], Gemini [25], dan Claude 3 [26]. Model-model terbaru ini telah menunjukkan peningkatan signifikan dalam memahami konteks kode yang panjang dan memberikan saran perbaikan yang lebih akurat. Selain itu, integrasi dengan infrastruktur seperti Groq LPU [27] dapat lebih dioptimalkan untuk mencapai latensi di bawah 1 detik untuk model berukuran 70B parameter.

Berdasarkan hasil penelitian dan keterbatasan yang ditemukan, maka beberapa saran untuk pengembangan selanjutnya adalah sebagai berikut:

1. Mengembangkan perbaikan pada fitur review seleksi kode agar dapat berjalan secara stabil dan akurat.
2. Menambahkan fitur pengujian pengguna (*user testing*) untuk memperoleh evaluasi dari sisi pengalaman pengguna (user experience).
3. Mengintegrasikan mekanisme penyimpanan riwayat hasil review sehingga pengguna dapat meninjau kembali hasil analisis sebelumnya.
4. Menambahkan fitur perbandingan hasil review antara model AI untuk membantu pengguna memilih model yang paling sesuai.
5. Mengembangkan dukungan terhadap lebih banyak bahasa pemrograman dan penambahan metode review yang lebih spesifik.

DAFTAR PUSTAKA

[1] Kohlbacher, S., et al. (2023). *Atoms of confusion in novice programmers' code: An empirical study*. IEEE Transactions on Software Engineering, 49(3), 1234–1248.

[2] Langhout, E., & Aniche, M. (2021). *An empirical study on code readability and error-proneness in novice programmers*. Journal of Systems and Software, 178, 110938.

[3] Thornhill, J., & Borg, M. (2022). *Code quality and its relation to software defects: A large-scale empirical study*. Empirical Software Engineering, 27(4), 1–29.

[4] Martin, R. C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Upper Saddle River, NJ: Prentice Hall.

[5] Vaswani, A., et al. (2017). *Attention is all you need*. Advances in Neural Information Processing Systems (NeurIPS), 30.

[6] Brown, T. B., et al. (2020). *Language models are few-shot learners*. Advances in Neural Information Processing Systems, 33, 1877–1901.

[7] Microsoft. (2024). *Visual Studio Code Extension API Documentation*. https://code.visualstudio.com/api

[8] Groq. (2024). *Groq API Documentation*. https://console.groq.com/docs

[9] OpenAI. (2023). *GPT-4 Technical Report*. https://arxiv.org/abs/2303.08774

[10] Meta AI. (2023). *LLaMA: Open and Efficient Foundation Language Models*. https://arxiv.org/abs/2302.13971

[11] D. Gopstein, J. Iannacone, Y. Yan, R. DeLine, and E. Koutsofios, "Understanding misunderstandings in source code," in *Proceedings of the 2017*

11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE), 2017, pp. 129–139.

[12] R. S. Pressman, *Software Engineering: A Practitioner's Approach*, 8th ed. New York, NY, USA: McGraw-Hill, 2014.

[13] I. Sommerville, *Software Engineering*, 10th ed. Boston, MA, USA: Pearson, 2016.

[14] ISO/IEC 25010:2011, *Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models*. ISO, 2011.

[15] M. Chen *et al*., "Evaluating large language models trained on code," *arXiv preprint* arXiv:2107.03374, 2021.

[16] B. Roziere *et al*., "Code Llama: Open foundation models for code," *arXiv preprint* arXiv:2308.12950, 2023.

[17] R. Li *et al*., "StarCoder: May the source be with you!," *arXiv preprint* arXiv:2305.06161, 2023.

[18] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *ACM Computing Surveys*, vol. 55, no. 9, pp. 1–35, 2023.

[19] J. White *et al*., "A prompt pattern catalog to enhance prompt engineering with ChatGPT," *arXiv preprint* arXiv:2302.11382, 2023.

[20] A. Robins, J. Rountree, and N. Rountree, "Learning and teaching programming: A review and discussion," *Computer Science Education*, vol. 13, no. 2, pp. 137–172, 2003.

[21] M. N. Kholid, P. I. Santosa, and A. N. Hidayanto, "A systematic literature review on the use of AI in programming education," *Education and Information*

Technologies, vol. 27, no. 8, pp. 11025–11057, 2022.

[22] M. M. Rahman and C. K. Roy, "A systematic review of automated code review," *Information and Software Technology*, vol. 134, p. 106543, 2021.

[23] A. Bacchelli and C. Bird, "Expectations, outcomes, and challenges of modern code review," in *Proceedings of the 2013 International Conference on Software Engineering (ICSE)*, 2013, pp. 712–721.

[24] D. Wang, L. Li, C. Zhu, and Y. Liu, "An empirical study on the effectiveness of automated code review tools," *Journal of Systems and Software*, vol. 179, p. 110992, 2021.

[25] Google, "Gemini: A family of highly capable multimodal models," *Google AI Technical Report*, 2024. [Online]. Available: https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf

[26] Anthropic, "The Claude 3 model family: Opus, Sonnet, Haiku," *Anthropic Technical Report*, 2024. [Online]. Available: https://www.anthropic.com/news/claude-3-family

[27] Groq, "Groq LPU: Language Processing Unit technical whitepaper," *Groq Documentation*, 2024. [Online]. Available: https://groq.com/lpu/

[28] H. Touvron *et al*., "LLaMA: Open and efficient foundation language models," *arXiv preprint* arXiv:2302.13971, 2023.

[29] A. Chowdhery *et al*., "PaLM: Scaling language modeling with pathways," *arXiv preprint* arXiv:2204.02311, 2022.

[30] P. Liang *et al*., "Holistic evaluation of language models," *Annals of the New York Academy of Sciences*, vol. 1525, no. 1, pp. 140–158, 2023.