Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

Analisis Dampak *ChatGPT* Sebagai Code Assistant Terhadap Kualitas Kode Mahasiswa Informatika UTS

Savitri Sunariany Utami[1*], Siska Atmawan Oktavia[1], Muhammad Alfan Habib[2]

[1] Rekayasa Sistem, Informatika, Universitas Teknologi Sumbawa, Indonesia
[2]Program Studi Teknik Informatika, STMIK Syaikh Zainuddin Nahdlatul Wathan Anjani, Lombok, Indonesia
Email: [1]savitriutami43@gmail.com, [2]siska.atmawan.oktavia@uts.ac.id, [3]alfanhabib20@gmail.com

siska.atmawan.oktavia@uts.ac.id

Abstrak— Perkembangan teknologi kecerdasan buatan (*Artificial Intelligence*) telah menghadirkan berbagai alat bantu dalam pengembangan perangkat lunak, salah satunya *ChatGPT*. Penelitian ini bertujuan menganalisis dampak penggunaan *ChatGPT* sebagai *code assistant* terhadap kualitas kode program mahasiswa Informatika Universitas Teknologi Sumbawa. Metode eksperimen kuantitatif digunakan dengan membandingkan penyelesaian tugas pemrograman secara manual dan dengan bantuan *ChatGPT*. Kualitas kode dianalisis menggunakan *SonarQube* berdasarkan metrik *Maintainability Index*, *Cyclomatic Complexity*, dan *Reliability* (*Bug Count*), kemudian diuji menggunakan analisis statistik. Hasil penelitian menunjukkan bahwa tidak terdapat perbedaan yang signifikan pada seluruh metrik yang diuji antara kode yang dibuat secara manual dan dengan bantuan *ChatGPT* ($p > 0{,}05$). Temuan ini menunjukkan bahwa penggunaan *ChatGPT* tidak mempengaruhi kualitas kode program pada penelitian ini, namun tetap berpotensi dimanfaatkan untuk meningkatkan efisiensi pengembangan program. Selain itu, penelitian ini memberikan bukti empiris mengenai pengaruh penggunaan ChatGPT terhadap kualitas kode program berdasarkan metrik analisis statis SonarQube. Hasil penelitian ini diharapkan dapat menjadi referensi bagi pendidik, mahasiswa, dan peneliti dalam mengevaluasi pemanfaatan ChatGPT sebagai *code assistant* pada pembelajaran maupun pengembangan perangkat lunak.

Kata Kunci: *ChatGPT*, *Code Assistant*, Kualitas Kode Program, *SonarQube*, Eksperimen Kuantitatif

Abstract— *The rapid advancement of Artificial Intelligence (AI) has introduced various tools that support software development, one of which is ChatGPT. This study aims to analyze the impact of using ChatGPT as a code assistant on the quality of program code produced by Informatics students at Universitas Teknologi Sumbawa. A quantitative experimental method was employed by comparing programming tasks completed manually and with the assistance of ChatGPT. Code quality was evaluated using SonarQube based on three metrics: Maintainability Index, Cyclomatic Complexity, and Reliability (Bug Count), followed by statistical analysis to examine differences between the two conditions. The results indicate that there were no significant differences across all evaluated metrics between manually written code and code generated with ChatGPT assistance ($p > 0.05$). These findings suggest that the use of ChatGPT did not affect code quality in this study; however, it still has the potential to improve the efficiency of software development. Furthermore, this study provides empirical evidence regarding the impact of using ChatGPT on code quality, based on SonarQube static analysis metrics. The findings are expected to serve as a reference for educators, students, and researchers in evaluating the use of ChatGPT as a code assistant in both learning and software development contexts.*

Keywords: *ChatGPT, Code Assistant, Code Quality, SonarQube, Quantitative Experiment*

1. PENDAHULUAN

Teknologi saat ini berkembang dengan cepat dan membawa banyak perubahan dalam berbagai bidang, seperti pendidikan dan pemrograman. Adanya *Artificial Intelligence* (AI) adalah salah satu kemajuan teknologi yang mulai banyak digunakan sebagai alat bantu dalam pembelajaran dan pekerjaan sehari-hari. *Artificial Intelligence* (AI) adalah sebuah mesin atau robot yang diprogram atau dibuat oleh manusia dengan kemampuan untuk melakukan sesuatu. Inti dari teknologi *Artificial Intelligence* (AI) adalah kecerdasan mesin yang memiliki kemampuan untuk berpikir dan memberikan respons yang mirip dengan manusia. Hampir semua aspek kehidupan manusia memiliki kecerdasan buatan manusia ini, mulai dari pendidikan, kesehatan, hiburan, dan lainnya.
Salah satunya adalah *ChatGPT*, yaitu sebuah sistem berbasis bahasa alami yang populer di kalangan *developer*. *ChatGPT* adalah *Artificial Intelligence* yang dikembangkan oleh *OpenAI* dengan memanfaatkan arsitektur *Generative Pre-trained Transformer* (GPT) [1]. Selain mampu membantu menghasilkan kode program secara otomatis, berbagai penelitian menunjukkan bahwa kualitas kode yang dihasilkan *ChatGPT* masih dipengaruhi oleh karakteristik tugas dan cara pengguna memberikan instruksi (*prompt*) [2]. Berdasarkan *Stack Overflow Developer Survey* pada tahun 2024, sebanyak 74% *developer* dari sekitar lebih dari 65.000 responden menggunakan *ChatGPT* sebagai *AI search and developer tools* untuk membantu pengguna menjawab pertanyaan, memberikan saran, dan bahkan membantu menulis kode program. Pemanfaatan *Artificial Intelligence* kini telah merambah dunia pendidikan terutama pada pembelajaran pemrograman, karena dianggap dapat meningkatkan efektivitas dan keoptimalan dalam proses belajar-mengajar[3]. Dalam pembelajaran pemrograman, kemampuan menghasilkan kode yang berkualitas, mudah dipahami, dan mudah dipelihara merupakan kompetensi penting bagi mahasiswa Informatika. Namun, banyak mahasiswa masih mengalami kesulitan dalam memahami algoritma dan menerapkan prinsip penulisan kode yang baik. Kehadiran *ChatGPT* menawarkan solusi yang dapat membantu proses penulisan dan perbaikan kode secara otomatis [4]. Oleh karena itu, penulisan kode yang rapi, terorganisir, dan mudah dipahami merupakan komponen penting dalam menjaga kualitas kode program [5]

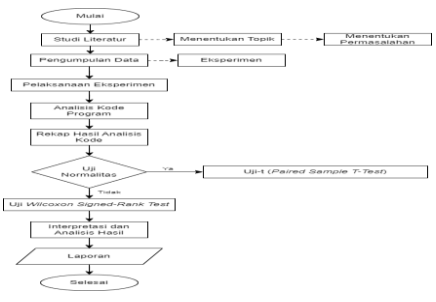
Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

Beberapa penelitian menunjukkan bahwa penggunaan *ChatGPT* dapat memberikan manfaat dalam pembelajaran pemrograman, seperti meningkatkan pemahaman konsep serta mempercepat penyelesaian tugas pemrograman. Namun, peningkatan efisiensi tersebut belum tentu diikuti oleh peningkatan kualitas kode program yang dihasilkan. Penelitian sebelumnya juga menunjukkan bahwa meskipun *ChatGPT* mampu menghasilkan kode yang benar secara fungsional, kualitas kode yang dihasilkan masih memerlukan evaluasi lebih lanjut dari aspek *maintainability*, kompleksitas, dan keandalan [6]. Penelitian Aldiansyah dan Sulistyo [7] menunjukkan bahwa kode yang dihasilkan dengan bantuan AI memiliki tingkat duplikasi yang lebih tinggi serta permasalahan *maintainability* yang lebih besar dibandingkan kode yang ditulis secara manual. Sementara itu, penelitian Idham dkk. [8] menyimpulkan bahwa kualitas kode yang dihasilkan *ChatGPT* sangat dipengaruhi oleh kualitas *prompt* yang diberikan pengguna dan tingkat kompleksitas permasalahan yang diselesaikan. Temuan-temuan tersebut menunjukkan bahwa hasil penggunaan *ChatGPT* dalam pemrograman masih bervariasi sehingga diperlukan penelitian lebih lanjut menggunakan pengukuran kualitas kode secara objektif. Meskipun penelitian sebelumnya telah mengkaji penggunaan *ChatGPT* dalam pembelajaran maupun pengembangan perangkat lunak, sebagian besar penelitian masih berfokus pada peningkatan produktivitas, efisiensi penyelesaian tugas, atau kualitas kode secara umum. Penelitian yang secara khusus mengevaluasi dampak penggunaan *ChatGPT* terhadap kualitas kode program mahasiswa menggunakan metrik analisis statis seperti *Maintainability Index*, *Cyclomatic Complexity*, dan *Reliability (Bug Count)* melalui *SonarQube* masih terbatas. Oleh karena itu, diperlukan penelitian yang mampu memberikan evaluasi empiris terhadap kualitas kode program yang dihasilkan dengan dan tanpa bantuan *ChatGPT*.
Berdasarkan kondisi tersebut, diperlukan penelitian yang mengkaji dampak penggunaan *ChatGPT* terhadap kualitas kode program secara objektif. Penelitian ini bertujuan menganalisis dampak penggunaan *ChatGPT* sebagai *code assistant* terhadap kualitas kode program mahasiswa Informatika Universitas Teknologi Sumbawa melalui pendekatan eksperimen kuantitatif. Analisis kualitas kode dilakukan menggunakan *SonarQube* berdasarkan tiga metrik, yaitu *Maintainability Index*, *Cyclomatic Complexity*, dan *Reliability (Bug Count)*. Ketiga metrik tersebut dipilih karena mewakili aspek keterpeliharaan, kompleksitas logika program, dan keandalan kode sehingga dapat memberikan gambaran yang lebih objektif mengenai kualitas kode program yang dihasilkan. *Maintainability Index* adalah metrik yang mengukur sebuah perangkat lunak atau kode program dari aspek tingkat kesulitan aplikasi tersebut untuk dilakukan perawatan dan pengembangan di masa akan datang [9]. Selanjutnya, *Cyclomatic complexity* adalah metrik yang digunakan untuk mengukur tingkat kompleksitas logika suatu program dengan menghitung jumlah jalur eksekusi independen yang ada dalam kode sumber. *Cyclomatic complexity* menghitung kompleksitas program dengan menggunakan jumlah jalur independen linear yang ada dalam kode program [10]. Terakhir, *bug* adalah kesalahan kode yang dapat menyebabkan sistem tidak berfungsi sebagaimana mestinya dan menyebabkan kegagalan operasional [11]. Jumlah cacat (*bug count*) yang ditemukan pada produk akhir biasanya digunakan untuk mengukur kualitas perangkat lunak [12].
Melalui pendekatan tersebut, penelitian ini memberikan bukti empiris mengenai pengaruh penggunaan *ChatGPT* terhadap kualitas kode program berdasarkan metrik analisis statis serta memperkaya referensi mengenai pemanfaatan *AI* sebagai *code assistant* dalam pembelajaran pemrograman.

2. METODOLOGI PENELITIAN

2.1 Tahapan Penelitian

Dalam penelitian ini, jenis penelitian yang dilakukan adalah eksperimen dengan pendekatan kuantitatif. Metode eksperimen dipilih karena penelitian ini bertujuan membandingkan kualitas kode program yang dihasilkan pada dua kondisi, yaitu tanpa bantuan *ChatGPT* dan dengan bantuan *ChatGPT*. Sementara itu, pendekatan kuantitatif digunakan karena hasil analisis kualitas kode dinyatakan dalam bentuk data numerik yang selanjutnya dianalisis menggunakan uji statistik untuk mengetahui ada atau tidaknya perbedaan yang signifikan antara kedua kondisi.Metode penelitian eksperimen adalah metode yang digunakan untuk mengetahui bagaimana perlakuan tertentu berdampak pada yang lain dalam situasi yang terkontrol [13]. Sedangkan pendekatan yang digunakan pada penelitian adalah pendekatan kuantitatif, yang di mana pendekatan adalah cara yang digunakan untuk menjawab pertanyaan dalam sebuah penelitian dengan memanfaatkan data dalam bentuk angka (numerik) [14]. Selanjutnya, *flowchart* digunakan untuk menunjukkan urutan langkah-langkah dalam penelitian [15]. Alur penelitian ini dapat dilihat pada di bawah ini:



Gambar 1. Diagram Alir Penelitian

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

Berdasarkan diagram alir penelitian pada Gambar 1, penelitian diawali dengan tahap studi literatur untuk mengumpulkan referensi dari buku dan jurnal ilmiah yang berkaitan dengan penelitian eksperimen, metode kuantitatif, penggunaan *ChatGPT* dalam pemrograman, serta analisis kualitas kode program. Hasil studi literatur digunakan sebagai dasar dalam menentukan topik penelitian dan merumuskan permasalahan penelitian. Selanjutnya, instrumen penelitian berupa tugas pemrograman divalidasi oleh ahli di bidang Teknologi Informasi untuk memastikan kesesuaian materi, tingkat kesulitan, kejelasan instruksi, dan kesesuaian instrumen dengan tujuan penelitian. Proses validasi ini dilakukan agar instrumen yang digunakan layak diterapkan pada pelaksanaan eksperimen serta menghasilkan data yang dapat dipercaya. Tahap berikutnya adalah pengumpulan data melalui pelaksanaan eksperimen terhadap partisipan penelitian. Eksperimen dilakukan dengan dua perlakuan pada kelompok yang sama, yaitu penyelesaian tugas pemrograman tanpa bantuan *ChatGPT* dan penyelesaian tugas pemrograman dengan bantuan *ChatGPT*. Kode program yang dihasilkan kemudian dianalisis menggunakan *SonarQube* untuk memperoleh nilai *Maintainability Index, Cyclomatic Complexity, dan Reliability (Bug Count)*. Ketiga parameter tersebut dipilih karena mewakili aspek utama kualitas kode program yang mampu memberikan evaluasi kualitas kode secara lebih komprehensif. *Maintainability Index* digunakan untuk mengevaluasi tingkat keterpeliharaan kode, *Cyclomatic Complexity* digunakan untuk mengukur kompleksitas logika program, sedangkan *Reliability (Bug Count)* digunakan untuk mengidentifikasi potensi kesalahan (*bug*) yang dapat memengaruhi keandalan perangkat lunak. Selanjutnya, hasil analisis direkapitulasi menggunakan spreadsheet sebelum dilakukan uji normalitas dengan metode *Shapiro–Wilk* pada perangkat lunak Jamovi. Berdasarkan hasil uji normalitas, data yang berdistribusi normal dianalisis menggunakan *Paired Sample T-Test*, sedangkan data yang tidak berdistribusi normal dianalisis menggunakan *Wilcoxon Signed-Rank Test*. Tahap akhir penelitian dilakukan dengan menginterpretasikan hasil pengujian statistik untuk menjawab rumusan masalah, kemudian menyusun kesimpulan berdasarkan hasil analisis yang diperoleh.

2.2 Metode Analisis Data

Analisis data dilakukan untuk mengetahui apakah terdapat perbedaan kualitas kode program antara pengerjaan tanpa bantuan *ChatGPT* dan dengan bantuan *ChatGPT*. Tahap analisis diawali dengan pengolahan data hasil analisis SonarQube untuk memperoleh nilai *Maintainability Index*, *Cyclomatic Complexity*, dan *Reliability (Bug Count)*. SonarQube dipilih karena mampu melakukan analisis statis secara otomatis terhadap kualitas kode program berdasarkan metrik yang objektif dan konsisten. Selanjutnya, hasil analisis direkapitulasi menggunakan Microsoft Excel dan diolah menggunakan perangkat lunak Jamovi sebagai dasar pengujian statistik. Sebelum dilakukan pengujian hipotesis, data diuji normalitas menggunakan metode Shapiro–Wilk karena jumlah sampel penelitian kurang dari 50. Data dinyatakan berdistribusi normal apabila nilai signifikansi (*p-value*) > 0,05, sedangkan *p-value* ≤ 0,05 menunjukkan bahwa data tidak berdistribusi normal. Pemilihan metode uji statistik didasarkan pada hasil uji normalitas. Apabila data berdistribusi normal, digunakan *Paired Sample t-Test* untuk menguji perbedaan antara dua kondisi yang saling berpasangan [15]. Sebaliknya, apabila data tidak berdistribusi normal, digunakan *Wilcoxon Signed-Rank Test* sebagai uji statistik nonparametrik untuk membandingkan dua kelompok data berpasangan [16]. Pengambilan keputusan didasarkan pada taraf signifikansi sebesar 5% (α = 0,05). Jika *p-value* < 0,05, maka terdapat perbedaan yang signifikan antara kedua perlakuan, sedangkan jika *p-value* ≥ 0,05, maka tidak terdapat perbedaan yang signifikan. Hasil pengujian statistik kemudian diinterpretasikan berdasarkan nilai *p-value* dan didukung dengan nilai *effect size* untuk mengetahui besarnya pengaruh penggunaan *ChatGPT* terhadap kualitas kode program. Penggunaan *effect size* bertujuan untuk melengkapi hasil uji signifikansi sehingga tidak hanya menunjukkan ada atau tidaknya perbedaan, tetapi juga besarnya pengaruh yang ditimbulkan. Interpretasi nilai *effect size* mengacu pada kategori sebagaimana ditunjukkan pada Tabel 1 [17].

Nilai *Effect Size*	Kategori *Effect Size*
0,2	*Small*
0,5	*Medium*
0,8	*Large*

Tabel 1. Kategori *Effect Size*

3. HASIL DAN PEMBAHASAN

3.1 Hasil Analisis *SonarQube*

SonarQube adalah sebuah *tools* yang dapat melakukan analisis statis terhadap berbagai bahasa pemrograman, membantu pengembang meningkatkan kualitas dan konsistensi kode dengan menemukan *bug*, kerentanan (*vulnerabilities*), dan *code smell* dalam kode program. Selain itu, *SonarQube* memiliki kemampuan untuk memeriksa berbagai proyek atau aplikasi sekaligus dalam satu alur kerja, mendukung proses pemantauan kualitas kode yang berkelanjutan [18]. *SonarQube* juga memiliki aturan analisis yang lebih lengkap, memberikan *dashboard* yang jelas untuk melihat kualitas kode secara menyeluruh, dan mendukung lebih banyak bahasa pemrograman [19].

Berdasarkan hasil analisis menggunakan *SonarQube*, diperoleh nilai kualitas kode program pada kedua kondisi eksperimen, yaitu tanpa bantuan *ChatGPT* dan dengan bantuan *ChatGPT* pada metrik yang diuji yaitu *maintainability index*, *cyclomatic complexity*, dan *reliability* (*bug count*). Berikut adalah tabel hasil rekapan analisis *SonarQube* yang direkap dalam *excel*:

No	Maintainability_NoGPT	Maintainability_ChatGPT	Complexity_NoGPT	Complexity_ChatGPT	Bug_NoGPT	Bug_ChatGPT
1	0	0	4	8	0	0
2	0	0	4	1	1	0

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

No	Maintainability_NoGPT	Maintainability_ChatGPT	Complexity_NoGPT	Complexity_ChatGPT	Bug_NoGPT	Bug_ChatGPT
3	0	0,8	4	20	0	0
4	0,7	0	6	1	0	0
5	0	0	4	1	0	0
6	0	0	4	4	0	0
7	0	0	8	2	0	0
8	0	0	4	9	0	0
9	0	0	7	11	0	0
10	0	0,7	4	8	0	0
11	0	0,3	4	9	0	0
12	0	0	4	1	0	0
13	0	0	10	3	0	0
14	1	0	9	3	0	0
15	0	-	4	-	0	-

Gambar 1. Rekapan Hasil Analisis *SonarQube*

Berdasarkan gambar di atas, kode program yang dibuat secara manual dan dengan bantuan *ChatGPT* menunjukkan perbedaan pada nilai *maintainability index* dan *cyclomatic complexity*. Namun, nilai *maintainability index* pada kedua kondisi cenderung serupa, sehingga tidak menunjukkan perbedaan yang berarti dalam kemudahan pemeliharaan kode. Sebaliknya, kode yang dihasilkan dengan bantuan *ChatGPT* memiliki nilai *cyclomatic complexity* yang lebih tinggi, yang mengindikasikan struktur logika program yang lebih kompleks. Pada aspek *reliability*, seluruh kode yang dianalisis memiliki jumlah *bug* sebesar nol. Temuan ini menunjukkan bahwa penggunaan *ChatGPT* tidak menghasilkan lebih banyak bug dibandingkan kode yang dibuat secara manual, sehingga tidak ditemukan kesalahan yang terdeteksi pada kedua kondisi penelitian.

3.2 Hasil Uji Statistik

Sebelum menentukan uji statistik yang digunakan, terlebih dahulu dilakukan uji normalitas pada data menggunakan uji *Shapiro-Wilk* karena data berjumlah kurang dari 50. Pengujian Statitik dilakukan menggunakan *tools* Jamovi. Jamovi adalah perangkat lunak statistik yang dapat diakses secara publik atau bersifat *open source* yang dibangun dengan statistik R, memungkinkan pengguna membuat modul, *library*, dan paket menggunakan *script* R [20]. Berikut adalah hasil uji normalitas dari ketiga metrik yang diuji menggunakan *tools*:

a. Maintainability Index

Berikut adalah hasil uji normalitas metrik *maintainability index*:

Normality Test (Shapiro-Wilk)

			W	p
Maintainability_ChatGPT	-	Maintainability_NoGPT	0.704	< .001

Note. A low p-value suggests a violation of the assumption of normality

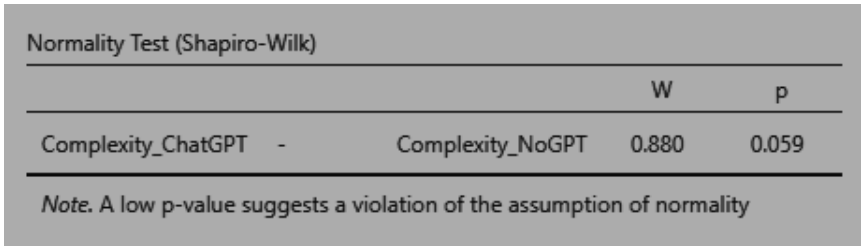
Gambar 1. Hasil Uji Normalitas *Maintainability Index*

Berdasarkan gambar 1, hasil pengujian menunjukkan bahwa data *maintainability index* tidak memenuhi asumsi data berdistribusi normal, dengan nilai signifikansi (*p-value*) yaitu kurang dari 0,001. Nilai tersebut lebih kecil dari 0,05. Oleh karena itu, data *maintainability index* ini tidak memenuhi asumsi untuk uji parametrik, sehingga digunakanlah uji non-parametrik *Wilcoxon Signed Rank-Test* yang tidak mensyaratkan data berdistribusi normal.

b. Cyclomatic Complexity

Berikut adalah hasil uji normalitas metrik *cyclomatic complexity*:

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS



Normality Test (Shapiro-Wilk)

			W	p
Complexity_ChatGPT	-	Complexity_NoGPT	0.880	0.059

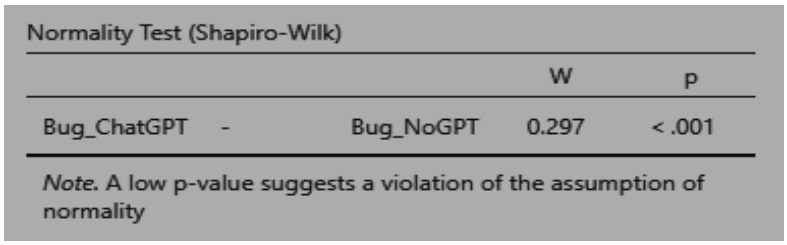
Note. A low p-value suggests a violation of the assumption of normality

Gambar 2. Hasil Uji Normalitas *Cyclomatic Complexity*

Berdasarkan gambar 2, hasil pengujian menunjukkan bahwa data *cyclomatic complexity* berdistribusi normal, dengan nilai signifikansi (*p-value*) yaitu 0,059, di mana nilai tersebut lebih besar dari 0,05. Oleh karena itu, data *cyclomatic complexity* memenuhi asumsi untuk uji parametrik, yaitu *Paired Sample T-Test*.

c. Reliability (*Bug Count*)

Berikut adalah hasil uji normalitas metrik *reliability (bug count)*:



Normality Test (Shapiro-Wilk)

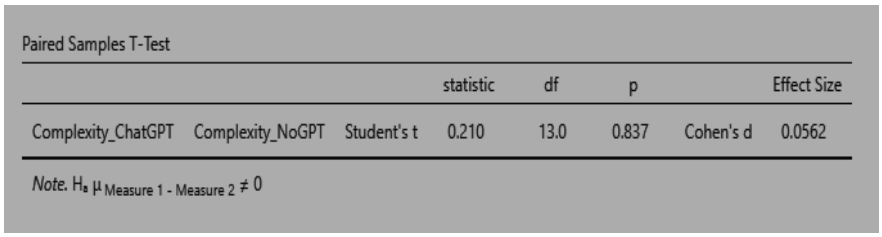
			W	p
Bug_ChatGPT	-	Bug_NoGPT	0.297	< .001

Note. A low p-value suggests a violation of the assumption of normality

Gambar 3. Hasil Uji Normalitas *Bug Count*

Berdasarkan gambar 3, hasil pengujian menunjukkan bahwa data *bug count* tidak memenuhi asumsi data berdistribusi normal, dengan nilai signifikansi (*p-value*) yaitu kurang dari 0,001. Nilai tersebut lebih kecil dari 0,05. Oleh karena itu, data *bug count* ini tidak memenuhi asumsi untuk uji parametrik, sehingga digunakanlah uji non-parametrik *Wilcoxon Signed Rank-Test* yang tidak mensyaratkan data berdistribusi normal.
Setelah mengetahui metrik yang diuji berdistribusi normal atau tidak, maka kemudian dilakukanlah uji statistik yang sesuai. Hasil pengujian statistik dari ketiga metrik adalah sebagai berikut:
a. Uji *Paired Sample T-Test*

Pengujian ini digunakan untuk mengetahui apakah terdapat perbedaan rata-rata yang signifikan antara dua kelompok data yang saling berpasangan. Walaupun menggunakan individu yang sama, peneliti tetap memperoleh 2 macam data sampel, yaitu data dari perlakuan pertama dan data dari perlakuan kedua yaitu nilai kualitas kode program yang dibuat secara manual dan yang menggunakan bantuan *ChatGPT* (Nuryadi, 2017). Berikut adalah hasil pengujian uji statistik *Paired Sample T-Test* pada data *cyclomatic complexity*:



Paired Samples T-Test

			statistic	df	p		Effect Size
Complexity_ChatGPT	Complexity_NoGPT	Student's t	0.210	13.0	0.837	Cohen's d	0.0562

Note. H_a $\mu_{\text{Measure 1 - Measure 2}} \neq 0$

Gambar 4. Uji Statistik *Cyclomatic Complexity*

Berdasarkan hasil pengujian data yang dilakukan dengan menggunakan Jamovi, diperoleh nilai signifikansi (*p-value*) untuk *cyclomatic complexity* adalah 0,837. Berdasarkan nilai *p-value* tersebut, hasil pengujian menunjukkan bahwa tidak terdapat perbedaan yang signifikan dari kualitas kode yang dihasilkan secara manual dan berbantuan *ChatGPT*. Selain itu, diperoleh juga nilai *effect size* (*Cohen's d*) sebesar 0,0562, di mana hal ini menunjukkan bahwa perbedaan yang terjadi pada kedua kondisi memiliki efek yang sangat kecil. Hal ini mengindikasikan bahwa penggunaan *ChatGPT* tidak memberikan pengaruh yang berarti terhadap kualitas kode program yang dihasilkan dari segi kompleksitas.

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

b. Uji *Wilcoxon Signed-Rank Test*

Pengujian ini adalah uji non-parametrik yang digunakan untuk menentukan perbedaan antara dua kelompok data berpasangan, terutama dalam kasus di mana data tidak memenuhi asumsi normalitas [16]. Dalam kasus ini, uji *Wilcoxon* digunakan sebagai alternatif dari Uji *Sampel Paired T-Test*.
Berdasarkan hasil uji normalitas yang dilakukan sebelumnya, diperoleh bahwa data *maintainability index* dan *reliability* (*bug count*) tidak memenuhi asumsi distribusi normal karena memiliki nilai signifikasi kurang dari 0,001 sehingga dilakukan uji *wilcoxon signed-rank test* ini sebagai bentuk validasi statistik terhadap hasil analisis sebelumnya mengenai kualitas kode yang dihasilkan secara manual dan kode yang dihasilkan berbantuan *ChatGPT*. Berikut adalah hasil pengujian statistik pada data *maintainability index* dan *bug count*:

Paired Samples T-Test

			Statistic	p		Effect Size
Maintainability_ChatGPT	Maintainability_NoGPT	Wilcoxon W	12.0[a]	0.281	Rank biserial correlation	0.600

Note. H_a $\mu_{\text{Measure 1 - Measure 2}} \neq 0$
[a] 9 pair(s) of values were tied

Gambar 5. Uji Statistik *Maintainability Index*

Paired Samples T-Test

			Statistic	p		Effect Size
Bug_ChatGPT	Bug_NoGPT	Wilcoxon W	0.00[a]	1.000	Rank biserial correlation	−1.00

Note. H_a $\mu_{\text{Measure 1 - Measure 2}} \neq 0$
[a] 13 pair(s) of values were tied

Gambar 6. Uji Statistik Bug Count

Berdasarkan hasil pengujian pada gambar 4.9 dan gambar 4.10, diperoleh nilai signifikansi (*p-value*) sebesar 0,281 dan 1.000. Hal ini menunjukkan bahwa tidak ditemukan adanya perbedaan yang signifikan antara kedua kode yang dihasilkan. Oleh karena itu, dapat disimpulkan bahwa penggunaan *ChatGPT* tidak memberikan pengaruh secara signifikan terhadap kualitas kode program yang dibuat oleh partisipan. Temuan ini sejalan dengan analisis sebelumnya yang menunjukkan bahwa kode yang dibuat secara manual dan kode yang dibuat dengan bantuan *ChatGPT* memiliki kualitas yang relatif sama pada metrik yang diuji.

3.3 Pembahasan

Berdasarkan hasil uji statistik, seluruh metrik kualitas kode menunjukkan nilai p-value > 0,05, yaitu *maintainability index* (p = 0,281; *effect size* = 0,600), *cyclomatic complexity* (p = 0,837; *effect size* = 0,0562), dan *bug count* (p = 1,000; *effect size* = -1,00). Hal ini menunjukkan bahwa penggunaan *ChatGPT* tidak memberikan pengaruh signifikan terhadap kualitas kode program dalam penelitian ini. Tidak signifikannya hasil ini dapat dipengaruhi oleh tingkat kompleksitas tugas yang relatif dasar serta telah divalidasi setara, yaitu program penentuan jenis bilangan dan program kasir. Kedua tugas hanya melibatkan struktur dasar pemrograman seperti input-output, percabangan, dan pengolahan data sederhana, sehingga dapat diselesaikan dengan baik baik secara manual maupun dengan bantuan *ChatGPT*. Selain itu, kemampuan dasar pemrograman partisipan yang cukup baik turut memengaruhi hasil, sehingga penggunaan *ChatGPT* lebih berperan sebagai alat bantu percepatan penulisan kode daripada peningkatan kualitas kode. Rendahnya jumlah *bug* pada kedua kondisi juga menunjukkan bahwa tingkat kesulitan tugas masih tergolong rendah, sehingga tidak menimbulkan perbedaan kualitas yang signifikan antara kedua metode. Meskipun demikian, terdapat perbedaan karakteristik penulisan kode. Kode manual menunjukkan variasi gaya penulisan, struktur, dan pendekatan penyelesaian, sedangkan kode berbantuan *ChatGPT* cenderung lebih konsisten dalam struktur, komentar, dan format output. Namun, perbedaan karakteristik tersebut tidak berpengaruh signifikan terhadap kualitas kode secara keseluruhan.
Hasil penelitian ini sejalan dengan penelitian Idham dkk. [8] yang menyatakan bahwa efektivitas *ChatGPT* dalam membantu pengembangan perangkat lunak sangat dipengaruhi oleh kualitas *prompt* yang diberikan pengguna dan tingkat kompleksitas permasalahan yang diselesaikan. Pada penelitian ini, tugas pemrograman yang diberikan masih berada pada tingkat kompleksitas dasar sehingga penggunaan *ChatGPT* tidak memberikan pengaruh yang signifikan terhadap kualitas kode program yang dihasilkan. Hal tersebut menunjukkan bahwa keberadaan *ChatGPT* tidak secara otomatis meningkatkan kualitas kode, tetapi dipengaruhi oleh konteks penggunaan dan karakteristik tugas yang dikerjakan. Di sisi lain, hasil penelitian ini berbeda dengan penelitian Aldiansyah dan Sulistyo [7] yang menemukan bahwa kode yang dihasilkan dengan bantuan *AI* memiliki tingkat duplikasi dan permasalahan *maintainability* yang lebih tinggi dibandingkan kode yang dibuat secara manual. Perbedaan hasil tersebut dipengaruhi oleh perbedaan objek penelitian, jenis tugas pemrograman, serta ruang lingkup evaluasi yang digunakan. Penelitian Aldiansyah dan Sulistyo berfokus pada

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

pengembangan aplikasi berbasis web dengan tingkat kompleksitas yang lebih tinggi, sedangkan penelitian ini menggunakan tugas pemrograman dasar pada mahasiswa sehingga karakteristik kode yang dihasilkan cenderung lebih sederhana.

Pendekatan eksperimen pada penelitian ini memiliki kelebihan karena memungkinkan perbandingan secara langsung antara kode yang dibuat tanpa bantuan *ChatGPT* dan dengan bantuan *ChatGPT* pada kelompok partisipan yang sama, sehingga dapat meminimalkan pengaruh perbedaan kemampuan antarpartisipan. Selain itu, penggunaan *SonarQube* memberikan evaluasi kualitas kode secara objektif berdasarkan metrik analisis statis. Namun demikian, penelitian ini masih memiliki keterbatasan, yaitu jumlah partisipan yang relatif terbatas karena partisipan dipilih berdasarkan kriteria tertentu, penggunaan tugas pemrograman dengan tingkat kompleksitas yang relatif dasar, serta evaluasi yang hanya menggunakan tiga metrik kualitas kode. Oleh karena itu, penelitian selanjutnya dapat melibatkan partisipan yang lebih beragam, menggunakan tugas pemrograman yang lebih kompleks, serta menambahkan metrik kualitas kode lainnya agar hasil penelitian menjadi lebih komprehensif.

4. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan mengenai analisis dampak penggunaan *ChatGPT* sebagai *code assistant* terhadap kualitas kode program, dapat disimpulkan bahwa penggunaan *ChatGPT* tidak memberikan pengaruh yang signifikan terhadap kualitas kode yang dihasilkan oleh partisipan. Analisis kualitas kode dilakukan menggunakan alat analisis dengan mengacu pada beberapa metrik, yaitu *maintainability index*, *cyclomatic complexity*, dan *reliability* yang diukur melalui jumlah *bug* (*bug count*). Hasil pengujian statistik menunjukkan bahwa seluruh metrik memiliki nilai signifikansi lebih besar dari 0,05, yaitu sebesar 0,281 pada *maintainability index*, 0,837 pada *cyclomatic complexity*, dan 1,000 pada *bug count*. Nilai tersebut menunjukkan bahwa tidak terdapat perbedaan yang signifikan antara kualitas kode yang ditulis secara manual dan kualitas kode yang dihasilkan dengan bantuan *ChatGPT*. Temuan ini mengindikasikan bahwa penggunaan *ChatGPT* sebagai *code assistant* belum mampu meningkatkan maupun menurunkan kualitas kode program secara signifikan pada konteks penelitian ini. Dengan kata lain, baik kode yang dibuat secara mandiri maupun kode yang disusun dengan bantuan *ChatGPT* memiliki tingkat keterpeliharaan, kompleksitas, dan reliabilitas yang relatif setara. Hasil tersebut menunjukkan bahwa *ChatGPT* dapat dimanfaatkan sebagai alat bantu dalam proses pemrograman tanpa memberikan dampak yang berarti terhadap kualitas kode yang dihasilkan berdasarkan metrik yang digunakan. Selain itu, tidak ditemukannya perbedaan yang signifikan juga dipengaruhi oleh beberapa faktor yang berkaitan dengan desain penelitian. Soal pemrograman yang diberikan kepada partisipan memiliki tingkat kesulitan yang relatif dasar dan setara sehingga tidak cukup kompleks untuk memperlihatkan perbedaan kualitas kode secara nyata. Di samping itu, partisipan yang merupakan mahasiswa Informatika telah memiliki kemampuan dasar pemrograman yang memadai, sehingga mereka mampu menyelesaikan tugas dengan baik baik secara manual maupun dengan bantuan *ChatGPT*. Oleh karena itu, hasil penelitian ini menunjukkan bahwa pada tugas pemrograman sederhana, penggunaan *ChatGPT* sebagai *code assistant* tidak memberikan dampak yang signifikan terhadap kualitas kode program yang dihasilkan. Penelitian ini memiliki beberapa keterbatasan, yaitu jumlah partisipan yang relatif terbatas karena dipilih berdasarkan kriteria tertentu, penggunaan tugas pemrograman dengan tingkat kompleksitas dasar, serta evaluasi kualitas kode yang hanya menggunakan tiga metrik, yaitu *Maintainability Index*, *Cyclomatic Complexity*, dan *Reliability (Bug Count)*. Oleh karena itu, penelitian selanjutnya disarankan melibatkan jumlah partisipan yang lebih banyak, menggunakan tugas pemrograman dengan tingkat kompleksitas yang lebih beragam, serta menambahkan metrik kualitas kode lainnya atau membandingkan beberapa *code assistant* berbasis *AI* agar diperoleh pemahaman yang lebih komprehensif mengenai pengaruh AI terhadap kualitas kode program.

UCAPAN TERIMAKASIH

Penulis mengucapkan terima kasih kepada Program Studi Informatika Universitas Teknologi Sumbawa atas dukungan selama pelaksanaan penelitian. Penulis juga menyampaikan apresiasi kepada seluruh mahasiswa yang telah bersedia menjadi partisipan dalam penelitian ini, serta kepada dosen pembimbing yang telah memberikan arahan, masukan, dan bimbingan selama proses penelitian hingga penyusunan artikel ini.

REFERENCES

[1] A. Bachtiar and R. Andrian, "The Impact of Chatgpt In Front-End Development With A Focus On Reactjs," *Jurnal Media Computer Science*, vol. 3, no. 1, pp. 79–82, 2024, doi: https://doi.org/10.37676/jmcs.v3i1.4361.

[2] Z. Liu, Y. Tang, X. Luo, Y. Zhou, and L. F. Zhang, "No Need to Lift a Finger Anymore? Assessing the Quality of Code Generation by ChatGPT," *IEEE Transactions on Software Engineering*, vol. 50, no. 6, pp. 1548–1584, 2024, doi: 10.1109/TSE.2024.3392499.

[3] A. N. Sihananto, P. W. Atmaja, and Sugiarto, "Pemanfaatan AI dalam Pembelajaran Pemrograman untuk Mahasiswa," *Seminar Nasional Rekayasa dan Teknologi*, vol. 4, no. 1, pp. 8–11, 2024, doi: 10.47970/snarstek.v2i1.706.

[4] M. M. Rachman, L. Wati, and M. K. Wardani, "Pemanfaatan Teknologi AI ChatGPT untuk Mendukung Pemahaman Dasar Pemrograman," *Citizen : Jurnal Ilmiah Multidisiplin Indonesia*, vol. 5, no. 4, pp. 1055–1063, 2025, doi: 10.53866/jimi.v5i4.948.

Management of Information System Journal
Vol 4, No 3, Juli 2026, Hal. 301 - 308
ISSN: 2964-9455 (media online)
DOI: 10.47065/mis.v2i3.3043
https://journal.fkpt.org/index.php/MIS

[5] F. I. Inayah and M. Idris, "Implementasi Clean Code pada Pengembangan Berbasis Web," *Prosiding Automata*, vol. 2, no. 2, pp. 113–116, 2021, [Online]. Available: https://journal.uii.ac.id/AUTOMATA/article/view/19528

[6] Y. Liu *et al.*, "Refining ChatGPT-Generated Code: Characterizing and Mitigating Code Quality Issues," *ACM Transactions on Software Engineering and Methodology*, vol. 33, no. 5, 2024, doi: 10.1145/3643674.

[7] Aldiansyah and D. Sulistyo, "Analisis Perbandingan Proses Pembangunan Aplikasi Berbasis Website: Pendekatan Manual Vs Ai-generated," *e-Proceeding of Engineering*, vol. 12, no. 1, pp. 2162–2169, 2025, [Online]. Available: https://repository.telkomuniversity.ac.id/home/catalog/id/217757/slug/analisis-perbandingan-proses-pembangunan-aplikasi-berbasis-website-pendekatan-manual-vs-ai-generated-dalam-bentuk-buku-karya-ilmiah.html%0A/home/catalog/id/217757/slug/analisis-perbandi

[8] Idham, A. Rahman, and M. Rizkillah, "ANALISIS KEEFEKTIFAN CHATGPT DALAM PERANCANGAN APLIKASI," *Jurnal Informatika Teknologi dan Sains (JINTEKS)*, vol. 6, no. 2, pp. 115–121, 2024, doi: https://doi.org/10.51401/jinteks.v6i2.4050.

[9] R. A. Wijaya and Karmilasari, "Pengukuran Kualitas Website Pengurus Cabang NU Depok Menggunakan Software Metric," *Jurnal Sisfokom (Sistem Informasi dan Komputer)*, vol. 10, no. 3, pp. 438–443, 2021, doi: 10.32736/sisfokom.v10i3.1267.

[10] M. I. N. Ilmi, Aminudin, and Z. Sari, "Dampak Test-Driven Development pada Kualitas Kode," *Jurnal Edukasi dan Penelitian Informatika (JEPIN)*, vol. 9, no. 3, pp. 371–377, 2023, [Online]. Available: https://jurnal.untan.ac.id/index.php/jepin/article/view/66815/75676600467

[11] A. Hardoni, "Integrasi SMOTE pada Naive Bayes dan Logistic Regression Berbasis Particle Swarm Optimization untuk Prediksi Cacat Perangkat Lunak," *Jurnal Sistem dan Teknologi Informasi (JUSTIN)*, vol. 9, no. 2, p. 144, 2021, doi: 10.26418/justin.v9i2.43173.

[12] E. H. A. Prastyo, Suhartono, M. Faisal, M. A. Yaqin, and R. A. J. Firdaus, "Naive Bayes Classification Untuk Prediksi Cacat Perangkat Lunak," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, vol. 9, no. 2, pp. 782–791, 2024, doi: https://doi.org/10.29100/jipi.v9i2.5508.

[13] Sugiyono, *Metodologi Penelitian Kuantitatif, Kualitatif dan R & D*. Bandung: Alfabeta Bandung, 2013.

[14] H. Sastrawan and H. Menap, *Fundamental Riset Kuantitatif*. Yogyakarta: Deepublish Digital, 2023.

[15] A. R. Barakbah, T. Karlita, and A. S. Ahsan, *Logika dan Algoritma*, no. tahun 1736. Politeknik Elektronika Negeri Surabaya, 2013.

[16] A. D. Wulansari, *Aplikasi Statistika Nonparametrik Dalam Penelitian*. Thalibul Ilmi Publishing & Education, 2023.

[17] L. Cohen, L. Manion, and K. Morrison, *Research Methods in Education*, 6th ed. New York, 2018. doi: 10.4324/9781315456539-19.

[18] M. Fadhli, Y. Fitrisia, and D. Nurmalasari, "Pengujian Kualitas Kode Program Aplikasi Bank Sampah DLHK Kota Pekanbaru Menggunakan Code Smell Tools," *Proceedings of the ASIL Annual Meeting*, vol. 10, no. 1, pp. 86–97, 2024, doi: https://doi.org/10.35143/jkt.v10i1.6211.

[19] M. Shen, A. Pillai, B. A. Yuan, J. C. Davis, and A. Machiry, "An Empirical Study on the Use of Static Analysis Tools in Open Source Embedded Software," vol. 1, no. 1, pp. 1–22, 2023, [Online]. Available: http://arxiv.org/abs/2310.00205

[20] A. B. Pratama, *Analisis Data Kuantitatif Dalam Penelitian Sosial Menggunakan Jamovi Konsep Dasar dan Aplikasinya*. Yogyakarta: Penerbit Gava Media, 2022.

[21] Nuryadi, T. D. Astuti, E. S. Utami, and M. Budiantara, *Dasar-dasar Statistik Penelitian*. Yogyakarta: Sibuku Media, 2017.