PENGEMBANGAN DEVELOPER PRODUCTIVITY INDEX BERBASIS METRIK REPOSITORI GITHUB

Popo Nugroho [1], Windu Gata[2], Irfan Wibowo[3], Dandy Herago Tantoko[4], Tungkot Siregar[5]

[1,2,3,4,5] Program Studi Ilmu Komputer, Fakultas Teknologi Informasi, Universitas Nusa Mandiri

Jl. Kramat Raya No. 18, Jakarta Pusat, DKI Jakarta
[1]14250016@nusamandiri.ac.id, [2]windu@nusamandiri.ac.id, [3]14250029@nusamandiri.ac.id,
[4]14250026@nusamandiri.ac.id, [5]14250003@nusamandiri.ac.id

Abstract

Developer productivity is an important factor in the sustainability of open source software projects. However, productivity measurement is often based on a single indicator, such as commits, pull requests, or code changes, which may not fully represent developer contributions. This study develops and evaluates a Developer Productivity Index (DPI) based on GitHub repository metrics. The DPI integrates four metrics: Commit Frequency (CF), Active Days (AD), Pull Request Participation (PRP), and Code Contribution (CC). Data were collected through the GitHub REST API from five open source repositories, namely React, TensorFlow, Kubernetes, Visual Studio Code, and Apache Spark, during January-December 2024. A total of 1,560 contributor accounts were analyzed, consisting of 1,554 human contributors and 6 bot accounts. The results show that bot accounts can affect productivity measurement, as two of the top three DPI scores in the overall dataset were generated by bot accounts. After bot separation, VS Code achieved the highest average human DPI score of 0.0337, while Tyriar obtained the highest individual DPI score of 0.8693. Distribution analysis indicates highly unequal contributions, with a Gini coefficient of 0.9636 for Code Contribution. Sensitivity analysis also shows stable DPI rankings, with Spearman correlations ranging from 0.9969 to 0.9996. The main contribution of this study is a reproducible composite index that combines activity intensity, consistency, collaborative participation, and code contribution while separating bot accounts from human contributors. Practically, the proposed DPI can help project maintainers and researchers evaluate contributor activity more proportionally than single-metric measurement.

Keywords : *Developer Productivity Index, GitHub Repository, Open Source Software, Repository Mining, Software Engineering Metrics.*

Abstrak

Produktivitas developer merupakan faktor penting dalam keberlanjutan proyek perangkat lunak open source. Namun, pengukuran produktivitas masih sering menggunakan indikator tunggal, seperti jumlah commit, pull request, atau perubahan kode, sehingga belum sepenuhnya merepresentasikan kontribusi developer. Penelitian ini mengembangkan dan mengevaluasi Developer Productivity Index (DPI) berbasis metrik repositori GitHub. DPI mengintegrasikan empat metrik, yaitu Commit Frequency (CF), Active Days (AD), Pull Request Participation (PRP), dan Code Contribution (CC). Data dikumpulkan melalui GitHub REST API dari lima repositori open source, yaitu React, TensorFlow, Kubernetes, Visual Studio Code, dan Apache Spark selama Januari-Desember 2024. Sebanyak 1.560 akun kontributor dianalisis, terdiri dari 1.554 human contributor dan 6 bot account. Hasil penelitian menunjukkan bahwa akun bot dapat memengaruhi pengukuran produktivitas karena dua dari tiga nilai DPI tertinggi pada seluruh dataset berasal dari akun bot. Setelah pemisahan bot, VS Code memperoleh rata-rata DPI human contributor tertinggi sebesar 0,0337, sedangkan Tyriar memperoleh DPI individu tertinggi sebesar 0,8693. Analisis distribusi menunjukkan kontribusi yang sangat timpang, dengan nilai Gini sebesar 0,9636 pada Code Contribution. Sensitivity analysis juga menunjukkan peringkat DPI yang stabil dengan korelasi Spearman antara 0,9969 hingga 0,9996. Kontribusi utama penelitian ini adalah menghasilkan indeks komposit yang dapat direproduksi untuk menggabungkan intensitas aktivitas, konsistensi kontribusi, partisipasi kolaboratif, dan volume perubahan kode dengan tetap memisahkan akun bot dari human contributor. Secara praktis, DPI dapat membantu pengelola proyek dan peneliti mengevaluasi aktivitas kontributor secara lebih proporsional dibandingkan pengukuran berbasis satu metrik.

Kata kunci : *Developer Productivity Index, Repositori GitHub, Open Source Software, Repository Mining*, Metrik Rekayasa Perangkat Lunak.

1. PENDAHULUAN

Produktivitas developer merupakan salah satu faktor penting dalam keberhasilan pengembangan perangkat lunak, terutama pada proyek open source yang melibatkan banyak kontributor dan aktivitas kolaboratif. Pada platform GitHub, aktivitas seperti commit, pull request, code review, issue tracking, dan continuous integration menghasilkan jejak digital yang dapat digunakan untuk menganalisis kontribusi developer secara objektif [1], [2], [3], [4], [5]. Data repositori tersebut juga telah banyak dimanfaatkan dalam penelitian software repository mining untuk memahami pola aktivitas developer, kualitas proses pengembangan, efektivitas kolaborasi, serta pemantauan proyek perangkat lunak secara multidimensi [4], [5].

Meskipun GitHub menyediakan berbagai metrik pengembangan perangkat lunak, pengukuran produktivitas developer masih menjadi tantangan. Produktivitas sering kali diukur menggunakan indikator tunggal, seperti jumlah commit, jumlah pull request, atau jumlah perubahan kode. Pendekatan tersebut belum sepenuhnya mampu menggambarkan produktivitas secara menyeluruh karena setiap metrik hanya merepresentasikan satu dimensi aktivitas. Developer dengan jumlah commit tinggi belum tentu memiliki kontribusi kode yang besar, sedangkan developer dengan perubahan kode besar belum tentu aktif secara konsisten atau terlibat dalam proses kolaborasi. Hal ini sejalan dengan berbagai kajian yang menekankan bahwa produktivitas developer bersifat multidimensi dan perlu dilihat dari kombinasi aktivitas, kolaborasi, kualitas proses, serta pengalaman developer [6], [7], [8].

Penelitian sebelumnya telah mengkaji berbagai aspek aktivitas developer pada repositori GitHub. Beberapa penelitian berfokus pada identifikasi keahlian teknis dan penambangan keterampilan pengembang dari aktivitas GitHub [9], [10]. Penelitian lain membahas dataset metrik developer serta pengukuran pengalaman kontributor dalam proyek open source [11]. Selain itu, aspek kolaborasi pengembangan juga telah dikaji melalui pull request, code review, perubahan metrik statis, dan waktu respons dalam proses kolaborasi [12], [13], [14], [15], [16]. Kontribusi developer juga dianalisis melalui pendekatan multidimensional data integration dan code ownership yang berkaitan dengan kualitas perangkat lunak [17], [18]. Faktor lain seperti software development waste, struktur repositori, serta praktik kolaborasi mikro dalam proyek open source juga diketahui berpengaruh terhadap efektivitas pengembangan perangkat lunak [19], [20], [21]. Perkembangan generative AI dan GitHub Copilot turut memperluas diskusi mengenai produktivitas developer dan kolaborasi pengembangan perangkat lunak modern [22], [23], [24], [25].

Namun, sebagian besar penelitian tersebut masih mengevaluasi metrik repositori secara terpisah atau berfokus pada aspek tertentu dari aktivitas pengembangan. Pengukuran berbasis satu metrik belum cukup untuk merepresentasikan produktivitas developer secara seimbang karena produktivitas pada proyek open source dipengaruhi oleh intensitas aktivitas, konsistensi kontribusi, partisipasi kolaboratif, dan volume perubahan kode. Selain itu, belum banyak penelitian yang mengintegrasikan beberapa metrik repositori GitHub ke dalam satu indeks kuantitatif yang dapat digunakan untuk membandingkan produktivitas kontributor antar proyek open source. Pengukuran produktivitas berbasis GitHub juga menghadapi tantangan keberadaan akun otomatis atau bot account. Pada proyek open source berskala besar, akun otomatis seperti dependency updater, continuous integration account, dan release automation account dapat menghasilkan aktivitas tinggi sehingga berpotensi memengaruhi interpretasi produktivitas developer manusia apabila tidak dipisahkan dari human contributor.

Berdasarkan gap tersebut, penelitian ini mengembangkan Developer Productivity Index (DPI) berbasis metrik repositori GitHub untuk mengevaluasi produktivitas kontributor pada proyek open source. DPI dibangun menggunakan empat metrik utama, yaitu Commit Frequency (CF), Active Days (AD), Pull Request Participation (PRP), dan Code Contribution (CC). Keempat metrik tersebut dipilih karena merepresentasikan dimensi berbeda dari aktivitas pengembangan perangkat lunak, yaitu intensitas aktivitas, konsistensi kontribusi, partisipasi kolaboratif, dan volume perubahan kode.

Kebaruan penelitian ini terletak pada pengembangan DPI sebagai indeks komposit yang menggabungkan beberapa metrik produktivitas

repositori GitHub secara seimbang. Selain itu, penelitian ini tidak hanya menghitung produktivitas seluruh akun, tetapi juga memisahkan hasil pengukuran antara bot account dan human contributor agar interpretasi produktivitas developer menjadi lebih representatif. Penelitian ini juga memvalidasi DPI dengan membandingkan hasil pemeringkatan terhadap metrik tunggal, menganalisis ketimpangan distribusi kontribusi menggunakan Pareto dan koefisien Gini, serta melakukan sensitivity analysis untuk melihat stabilitas DPI terhadap perubahan bobot metrik. Dengan demikian, DPI diharapkan dapat menjadi pendekatan alternatif yang lebih komprehensif, terukur, dan mudah direproduksi dalam mengevaluasi produktivitas kontributor pada proyek open source berbasis GitHub.

Penelitian ini menggunakan data dari lima repositori open source populer, yaitu React, TensorFlow, Kubernetes, Visual Studio Code, dan Apache Spark selama periode 1 Januari 2024 sampai dengan 31 Desember 2024. Data dikumpulkan melalui GitHub REST API dan dianalisis menggunakan pendekatan software repository mining. Tujuan penelitian ini adalah mengembangkan DPI, mengevaluasi pengaruh pemisahan akun bot terhadap hasil pengukuran produktivitas, membandingkan DPI dengan metrik tunggal, serta menganalisis distribusi kontribusi pada proyek open source.

2. METODOLOGI PENELITIAN

Penelitian ini menggunakan pendekatan kuantitatif berbasis software repository mining untuk mengembangkan dan mengevaluasi Developer Productivity Index (DPI) sebagai indeks pengukuran produktivitas kontributor pada proyek open source. Data penelitian diperoleh dari aktivitas repositori GitHub yang dapat diakses secara publik melalui GitHub REST API. Pendekatan ini digunakan karena aktivitas pengembangan perangkat lunak seperti commit, pull request, dan perubahan kode dapat diukur secara objektif dan direproduksi.

2.1. Skema Alur Penelitian

Alur penelitian dimulai dari studi pustaka, pemilihan repositori, pengumpulan data GitHub, pembersihan data, identifikasi akun bot dan human contributor, ekstraksi metrik produktivitas, normalisasi data, perhitungan DPI, validasi DPI terhadap metrik tunggal, analisis distribusi kontribusi, sensitivity analysis, dan penarikan kesimpulan.

Tahapan penelitian tersebut digunakan untuk memastikan bahwa DPI tidak hanya dihitung berdasarkan satu indikator tunggal, tetapi menggabungkan beberapa dimensi aktivitas pengembangan perangkat lunak. Alur penelitian dapat digambarkan pada Gambar 1.



Gambar 1. Diagram Alir Penelitian

2.2. Pengumpulan Data

Data penelitian dikumpulkan dari lima repositori open source populer di GitHub, yaitu React, TensorFlow, Kubernetes, Visual Studio Code (VS Code), dan Apache Spark. Pemilihan repositori dilakukan berdasarkan beberapa kriteria, yaitu bersifat open source, aktif dikembangkan, memiliki jumlah kontributor yang memadai, serta menyediakan data commit dan pull request yang dapat diakses secara publik.

Periode pengamatan ditetapkan mulai 1 Januari 2024 sampai dengan 31 Desember 2024. Pengumpulan data dilakukan menggunakan GitHub REST API dengan bantuan pustaka PyGithub pada bahasa pemrograman Python. Data yang dikumpulkan meliputi nama akun kontributor, nama repositori, jumlah commit, tanggal commit, jumlah pull request, jumlah baris kode yang ditambahkan, dan jumlah baris kode yang dihapus. Berdasarkan proses pengumpulan

data, diperoleh 1.560 akun kontributor dari lima repositori yang dianalisis.

2.3. Metrik Produktivitas Developer

Penelitian ini menggunakan empat metrik utama untuk menyusun Developer Productivity Index, yaitu Commit Frequency (CF), Active Days (AD), Pull Request Participation (PRP), dan Code Contribution (CC). Commit Frequency digunakan untuk mengukur intensitas aktivitas pengembangan berdasarkan jumlah commit yang dilakukan kontributor. Active Days digunakan untuk mengukur konsistensi aktivitas berdasarkan jumlah hari unik ketika kontributor melakukan commit. Pull Request Participation digunakan untuk mengukur keterlibatan kontributor dalam proses kolaborasi dan integrasi kode. Code Contribution digunakan untuk mengukur besarnya perubahan kode yang dilakukan oleh kontributor.

Code Contribution dihitung berdasarkan jumlah baris kode yang ditambahkan dan dihapus selama periode penelitian. Perhitungan Code Contribution ditunjukkan pada Persamaan (1).

$$CC = AddedLines + DeletedLines \quad (1)$$

Metrik Code Contribution digunakan untuk merepresentasikan volume perubahan kode, tetapi tidak dimaknai secara langsung sebagai kualitas kode. Oleh karena itu, metrik ini digabungkan dengan CF, AD, dan PRP agar pengukuran produktivitas tidak hanya bergantung pada jumlah perubahan kode.

2.4. Normalisasi dan Perhitungan DPI

Keempat metrik yang digunakan memiliki rentang nilai yang berbeda, sehingga perlu dilakukan normalisasi sebelum digabungkan ke dalam satu indeks. Normalisasi dilakukan menggunakan metode Min-Max Normalization agar seluruh metrik berada pada rentang nilai 0 sampai 1. Normalisasi diperlukan agar setiap metrik berada pada skala yang sebanding, sehingga tidak ada metrik dengan nilai numerik lebih besar yang mendominasi pembentukan indeks. Pendekatan serupa juga digunakan pada penelitian sebelumnya, di mana Min-Max Scaling diterapkan untuk mentransformasikan fitur ke dalam rentang 0 hingga 1 agar skala data menjadi lebih seragam dan stabil sebelum proses pemodelan [26]. Rumus normalisasi ditunjukkan pada Persamaan (2).

$$X' = (X - Xmin) / (Xmax - Xmin) \quad (2)$$

Keterangan:

X' adalah nilai hasil normalisasi, X adalah nilai asli metrik, Xmin adalah nilai minimum pada metrik, dan Xmax adalah nilai maksimum pada metrik.

Setelah proses normalisasi dilakukan, nilai CF, AD, PRP, dan CC digabungkan menjadi Developer Productivity Index menggunakan pendekatan equal-weight. Setiap metrik diberi bobot yang sama sebesar 0,25. Rumus DPI ditunjukkan pada Persamaan (3).

$$DPI = 0{,}25(CF') + 0{,}25(AD') + 0{,}25(PRP') + 0{,}25(CC') \quad (3)$$

Pendekatan equal-weight digunakan sebagai baseline objektif karena penelitian ini tidak menggunakan survei, kuesioner, atau penilaian pakar dalam menentukan bobot setiap metrik. Pemilihan bobot yang sama dilakukan karena keempat metrik merepresentasikan dimensi produktivitas yang berbeda dan saling melengkapi, yaitu intensitas aktivitas, konsistensi kontribusi, partisipasi kolaboratif, dan volume perubahan kode. Dengan memberikan bobot yang seimbang, penelitian ini menghindari dominasi salah satu metrik tertentu dalam pembentukan indeks. Selanjutnya, stabilitas pendekatan equal-weight diuji melalui sensitivity analysis untuk mengetahui apakah perubahan bobot metrik memengaruhi hasil pemeringkatan DPI. Normalisasi dan perhitungan DPI dilakukan dalam dua bentuk, yaitu DPI seluruh akun dan DPI khusus human contributor.

2.5. Identifikasi Akun Bot, Validasi, dan Analisa Data

Pada repositori open source berskala besar, sebagian aktivitas dapat berasal dari akun otomatis seperti bot, robot, dependency updater, continuous integration account, atau release automation account. Oleh karena itu, penelitian ini memisahkan akun otomatis dari human contributor agar hasil pengukuran produktivitas developer lebih representatif. Pemisahan akun otomatis diperlukan karena aktivitas yang dihasilkan oleh bot dapat membentuk pola perilaku yang berbeda dari pengguna manusia dan berpotensi memengaruhi interpretasi hasil analisis. Penelitian sebelumnya pada data log web server menunjukkan bahwa analisis berbasis perilaku dapat digunakan untuk mengidentifikasi aktivitas tidak normal, termasuk pola akses otomatis seperti bot crawling dan scanning [27].

Identifikasi akun bot dilakukan berdasarkan pola nama akun yang mengindikasikan otomatisasi, seperti bot, robot, github-actions, dependabot, renovate, gardener, ci, dan release-robot. Akun yang memenuhi pola tersebut dikategorikan sebagai bot account, sedangkan akun lainnya dikategorikan sebagai human

contributor. Hasil penelitian kemudian disajikan dalam dua bentuk, yaitu hasil DPI seluruh akun dan hasil DPI human contributor.

Validasi DPI dilakukan dengan membandingkan peringkat kontributor berdasarkan DPI terhadap peringkat berdasarkan metrik tunggal, yaitu Commit Frequency, Active Days, Pull Request Participation, dan Code Contribution. Perbandingan ini dilakukan untuk melihat apakah DPI memberikan hasil yang lebih seimbang dibandingkan penggunaan satu indikator saja.

Selain itu, dilakukan analisis distribusi menggunakan statistik deskriptif, Pareto, dan koefisien Gini untuk melihat tingkat ketimpangan kontribusi antar kontributor. Koefisien Gini digunakan karena mampu menunjukkan apakah kontribusi tersebar merata atau hanya didominasi oleh sebagian kecil kontributor inti. Nilai Gini mendekati 0 menunjukkan distribusi yang lebih merata, sedangkan nilai mendekati 1 menunjukkan distribusi yang semakin timpang.

Sensitivity analysis juga dilakukan untuk menguji stabilitas DPI terhadap perubahan bobot metrik. Pengujian dilakukan melalui beberapa skenario pembobotan, yaitu equal-weight, activity-oriented, collaboration-oriented, code-oriented, dan balanced-human. Hasil sensitivity analysis digunakan untuk melihat apakah perubahan bobot menyebabkan perubahan peringkat yang signifikan atau tidak. Seluruh hasil analisa digunakan untuk mengevaluasi apakah DPI dapat menjadi pendekatan alternatif dalam mengukur produktivitas developer pada proyek open source.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Pengumpulan Data

Pengumpulan data dilakukan pada lima repositori open source GitHub, yaitu React, TensorFlow, Kubernetes, Visual Studio Code (VS Code), dan Apache Spark selama periode 1 Januari 2024 sampai dengan 31 Desember 2024. Berdasarkan proses pengumpulan data melalui GitHub REST API, diperoleh 1.560 akun kontributor yang terdiri dari 1.554 human contributor dan 6 bot account. Total aktivitas yang berhasil dikumpulkan meliputi 38.894 commit, 19.479 pull request, dan 39.593.522 code contribution yang dihitung dari jumlah baris kode yang ditambahkan dan dihapus.

Ringkasan data repositori ditunjukkan pada Tabel 1.

TABEL I. RINGKASAN DATA REPOSITORI

Repo	Akun	Human	Bot	Cmt	PR	CC	DPI
React	104	103	1	2.161	2.105	1.553.728	0,0206
TensorFlow	472	470	2	15.790	363	17.431.117	0,0197
Kubernetes	458	456	2	6.994	4.071	7.366.215	0,0109
VS Code	272	271	1	10.281	8.534	12.463.357	0,0337
Apache Spark	254	254	0	3.668	4.406	779.105	0,0190

Berdasarkan Tabel 1, TensorFlow memiliki jumlah commit (Cmt) dan code contribution (CC) terbesar, sedangkan VS Code memiliki jumlah pull request (PR) tertinggi. Namun, repositori dengan rata-rata DPI human contributor tertinggi adalah VS Code sebesar 0,0337. Temuan ini mengindikasikan bahwa jumlah commit atau volume perubahan kode yang besar belum tentu menghasilkan nilai produktivitas rata-rata yang paling tinggi. DPI mempertimbangkan beberapa dimensi secara bersamaan, yaitu intensitas aktivitas, konsistensi kontribusi, partisipasi pull request, dan volume perubahan kode.

3.2 Identifikasi Akun Bot dan Human Contributor

Pada repositori open source berskala besar, sebagian aktivitas pengembangan dapat dilakukan oleh akun otomatis. Oleh karena itu, penelitian ini memisahkan akun bot dari human contributor agar hasil pengukuran produktivitas developer tidak bias. Berdasarkan pola username yang mengindikasikan otomatisasi, diperoleh 6 akun bot sebagaimana ditunjukkan pada Tabel 2.

TABEL II. AKUN BOT YANG TERIDENTIFIKASI.

No	Repo	Akun Bot	Commit	AD	PR	CC
1	Kubernetes	k8s-ci-robot	2.685	310	0	3.409.172
2	TensorFlow	tensorflower-gardener	5.670	362	0	892.144
3	React	dependabot [bot]	27	11	69	1.095
4	VS Code	dependabot [bot]	30	26	41	567
5	TensorFlow	dependabot [bot]	31	16	38	436
6	Kubernetes	k8s-release-robot	67	38	0	19.442

Hasil identifikasi menunjukkan bahwa meskipun jumlah akun bot hanya 6 dari 1.560 akun, beberapa bot memiliki aktivitas yang sangat tinggi. Akun tensorflower-gardener memiliki 5.670 commit dan aktif selama 362 hari, sedangkan k8s-ci-robot memiliki code contribution sebesar 3.409.172. Kondisi ini menunjukkan bahwa akun otomatis dapat memengaruhi hasil pengukuran DPI apabila tidak dipisahkan dari human contributor.

3.3 Hasil Pengukuran DPI Seluruh Akun

Tahap pertama perhitungan DPI dilakukan pada seluruh akun, baik human

contributor maupun bot account. Hasil lima akun dengan nilai DPI tertinggi pada seluruh dataset ditunjukkan pada Tabel 3.

TABEL III. LIMA AKUN DENGAN DPI TERTINGGI PADA SELURUH AKUN

Rank	Repo	Akun	Jenis Akun	DPI
1	VS Code	Tyriar	Human	0,6048
2	TensorFlow	tensorflower-gardener	Bot	0,5514
3	Kubernetes	k8s-ci-robot	Bot	0,5289
4	VS Code	jrieken	Human	0,4505
5	VS Code	roblourens	Human	0,3816

Berdasarkan Tabel 3, dua dari tiga akun dengan DPI tertinggi merupakan akun bot. Hal ini mengindikasikan bahwa aktivitas otomatisasi dapat menghasilkan nilai DPI tinggi karena memiliki commit, active days, atau code contribution yang besar. Oleh karena itu, hasil DPI seluruh akun lebih tepat digunakan untuk menggambarkan aktivitas repositori secara keseluruhan, bukan untuk menilai produktivitas developer manusia secara langsung.

3.4 Hasil Pengukuran DPI Human Contributor

Setelah akun bot dipisahkan, normalisasi dan perhitungan DPI dilakukan ulang khusus pada human contributor. Perhitungan ulang ini dilakukan agar nilai DPI human contributor tidak dipengaruhi oleh nilai ekstrem dari akun bot. Sepuluh human contributor dengan nilai DPI tertinggi ditunjukkan pada Tabel 4.

TABEL IV. SEPULUH HUMAN CONTRIBUTOR DENGAN DPI TERTINGGI

Rank	Repo	Akun	Cmt	AD	PR	CC	DPI
1	VS Code	Tyriar	1.826	210	513	3.446.506	0,8693
2	VS Code	jrieken	781	186	645	1.050.729	0,6070
3	VS Code	roblourens	561	219	536	293.367	0,5326
4	VS Code	sandy081	478	182	484	162.382	0,4529
5	VS Code	mjbvz	475	171	403	866.439	0,4519
6	VS Code	meganrogge	815	164	346	375.715	0,4421
7	VS Code	lszomoru	463	161	472	31.262	0,4150
8	Apache Spark	zhengruifeng	360	200	381	42.243	0,4142
9	Apache Spark	panbingkun	322	189	433	30.543	0,4140
10	Apache Spark	dongjoon-hyun	364	165	429	57.438	0,3924

Hasil pada Tabel 4 menunjukkan bahwa akun Tyriar dari repositori VS Code memiliki DPI tertinggi sebesar 0,8693. Nilai tersebut diperoleh karena akun tersebut memiliki kombinasi aktivitas yang tinggi pada beberapa metrik sekaligus, yaitu commit (Cmt), active days, pull request, dan code contribution. Hal ini berbeda dengan akun yang hanya unggul pada satu metrik tertentu, tetapi tidak selalu memiliki nilai DPI tertinggi. Dengan demikian, DPI dapat memberikan pemeringkatan yang lebih seimbang karena menggabungkan beberapa indikator produktivitas secara bersamaan.

3.5 Validasi DPI terhadap Metrik Tunggal

Validasi DPI dilakukan dengan membandingkan peringkat DPI terhadap peringkat berdasarkan metrik tunggal, yaitu Commit Frequency, Active Days, Pull Request Participation, dan Code Contribution. Hasil korelasi Spearman antara DPI dan setiap metrik tunggal ditunjukkan pada Tabel 5.

TABEL V. KORELASI SPEARMAN DPI TERHADAP METRIK TUNGGAL

Perbandingan	**Spearman Rho**
DPI vs Commit Frequency	0,9409
DPI vs Active Days	0,9421
DPI vs Pull Request Participation	0,4648
DPI vs Code Contribution	0,8287

Berdasarkan Tabel 5, DPI memiliki korelasi tinggi dengan Commit Frequency dan Active Days, yaitu 0,9409 dan 0,9421. Hal ini menunjukkan bahwa DPI tetap menangkap dimensi intensitas dan konsistensi aktivitas developer. Korelasi DPI dengan Code Contribution juga cukup tinggi sebesar 0,8287, sedangkan korelasi dengan Pull Request Participation lebih rendah sebesar 0,4648. Kondisi ini mengindikasikan bahwa DPI tidak hanya mengikuti pola satu metrik tertentu, tetapi menggabungkan beberapa dimensi produktivitas secara bersamaan.

Perbandingan peringkat juga menunjukkan bahwa penggunaan metrik tunggal dapat menghasilkan interpretasi yang berbeda. Misalnya, akun Flamefire memiliki code contribution tertinggi sebesar 4.336.043, tetapi tidak menempati peringkat tertinggi DPI karena nilai commit, active days, dan pull request tidak setinggi akun lain. Sebaliknya, akun Tyriar menempati peringkat tertinggi DPI karena memiliki performa yang relatif kuat pada beberapa metrik sekaligus. Hal ini menunjukkan bahwa DPI memberikan penilaian yang lebih seimbang dibandingkan penggunaan satu indikator tunggal.

3.6 Analisis Distribusi Kontribusi

Analisis distribusi dilakukan pada dataset human contributor untuk melihat pola ketimpangan kontribusi antar developer. Statistik deskriptif dan koefisien Gini ditunjukkan pada Tabel 6.

TABEL VI. STATISTIK DISTRIBUSI HUMAN CONTRIBUTOR

Variabel	Mean	Median	Max	Gini
Commit Frequency	19,55	2	1.826	0,8333
Active Days	10,82	2	219	0,7539
Pull Request Participation	12,44	1	709	0,8772
Code Contribution	22.696,70	152,5	4.336.043	0,9636
DPI	0,0195	0,0020	0,8693	0,8308

Berdasarkan Tabel 6, seluruh variabel memiliki nilai mean yang jauh lebih besar daripada median. Kondisi ini menunjukkan bahwa distribusi kontribusi tidak merata dan cenderung dipengaruhi oleh sejumlah kecil kontributor dengan aktivitas sangat tinggi. Nilai Gini tertinggi terdapat pada Code Contribution sebesar 0,9636, yang mengindikasikan ketimpangan sangat tinggi pada volume perubahan kode.

Hasil analisis Pareto memperkuat temuan tersebut. Pada metrik Code Contribution, 1% kontributor teratas menghasilkan 66,87% dari total code contribution, 5% kontributor teratas menghasilkan 91,50%, dan 10% kontributor teratas menghasilkan 96,10%. Pada metrik DPI, 10% kontributor teratas menyumbang 72,63% dari total nilai DPI. Temuan ini mengindikasikan bahwa produktivitas pada proyek open source cenderung terkonsentrasi pada sejumlah kecil core contributor.

3.7 Sensitivity Analysis

Sensitivity analysis dilakukan untuk menguji stabilitas DPI terhadap perubahan bobot metrik. Lima skenario pembobotan digunakan, yaitu equal-weight, activity-oriented, collaboration-oriented, code-oriented, dan balanced-human. Hasil perbandingan setiap skenario terhadap skenario equal-weight ditunjukkan pada Tabel 7.

TABEL VII. HASIL SENSITIVITY ANALYSIS

Skenario	Spearman terhadap Equal-weight	Top-10 Overlap
Activity-oriented	0,9983	90%
Collaboration-oriented	0,9969	100%
Code-oriented	0,9990	80%
Balanced-human	0,9996	100%

Berdasarkan Tabel 7, seluruh skenario memiliki korelasi Spearman yang sangat tinggi terhadap skenario equal-weight, yaitu berada pada rentang 0,9969 sampai 0,9996. Selain itu, overlap sepuluh peringkat teratas berada pada rentang 80% sampai 100%. Hasil ini menunjukkan bahwa pemeringkatan DPI relatif stabil terhadap perubahan bobot metrik.

Meskipun demikian, beberapa akun mengalami perubahan peringkat pada skenario tertentu. Misalnya, akun Flamefire naik pada skenario code-oriented karena memiliki code contribution terbesar. Sebaliknya, akun dengan aktivitas pull request tinggi cenderung lebih menonjol pada skenario collaboration-oriented. Hal ini mengindikasikan bahwa perubahan bobot dapat memengaruhi posisi akun tertentu, terutama akun yang sangat dominan pada satu dimensi produktivitas.

3.8 Pembahasan

Hasil penelitian menunjukkan bahwa pengukuran produktivitas developer pada proyek open source perlu dilakukan secara multidimensi. Penggunaan satu metrik tunggal seperti jumlah commit atau jumlah baris kode dapat menghasilkan interpretasi yang kurang seimbang. Akun dengan code contribution besar belum tentu aktif secara konsisten, sedangkan akun dengan pull request tinggi belum tentu memiliki volume perubahan kode yang besar. Oleh karena itu, integrasi beberapa metrik ke dalam DPI memberikan pendekatan alternatif yang lebih komprehensif.

Temuan ini sejalan dengan penelitian sebelumnya yang menyatakan bahwa produktivitas developer tidak cukup diukur menggunakan satu indikator tunggal, tetapi perlu dilihat dari kombinasi aktivitas, kolaborasi, kualitas proses, dan pengalaman developer [6], [7], [8]. Berbeda dengan penelitian tersebut yang lebih menekankan konsep produktivitas dan pengalaman developer secara umum, penelitian ini mengembangkan pendekatan kuantitatif melalui Developer Productivity Index (DPI) yang menggabungkan Commit Frequency, Active Days, Pull Request Participation, dan Code Contribution. Dengan demikian, DPI memberikan bentuk pengukuran yang lebih operasional karena setiap dimensi produktivitas diterjemahkan ke dalam metrik repositori GitHub yang dapat dihitung dan direproduksi.

Pemisahan akun bot dan human contributor menjadi bagian penting dalam penelitian ini. Hasil DPI seluruh akun menunjukkan bahwa dua akun bot masuk dalam

tiga peringkat tertinggi. Hal ini mengindikasikan bahwa aktivitas otomatisasi memiliki pengaruh besar terhadap metrik repositori GitHub. Jika akun bot tidak dipisahkan, interpretasi produktivitas developer manusia dapat menjadi bias. Setelah akun bot dipisahkan dan DPI human contributor dihitung ulang, hasil pemeringkatan menjadi lebih representatif untuk mengevaluasi aktivitas developer manusia.

Hasil penelitian ini juga melengkapi penelitian terdahulu mengenai kontribusi developer berbasis data repositori. Penelitian sebelumnya telah mengukur kontribusi developer melalui integrasi data multidimensi dan code ownership yang berkaitan dengan kualitas perangkat lunak [17], [18]. Namun, penelitian ini menambahkan aspek pemisahan akun bot dan human contributor agar hasil pengukuran produktivitas tidak bias oleh aktivitas otomatis. Selain itu, penggunaan analisis distribusi dan sensitivity analysis memperkuat evaluasi DPI karena tidak hanya menghasilkan peringkat kontributor, tetapi juga menunjukkan ketimpangan kontribusi dan stabilitas indeks terhadap perubahan bobot metrik.

Analisis distribusi menunjukkan bahwa kontribusi dalam proyek open source tidak tersebar merata. Nilai Gini yang tinggi pada seluruh metrik, khususnya Code Contribution sebesar 0,9636, menunjukkan adanya ketimpangan kontribusi yang kuat. Fenomena ini sejalan dengan karakteristik umum proyek open source, di mana sebagian besar aktivitas pengembangan sering kali dilakukan oleh sejumlah kecil core contributor. Dengan demikian, DPI tidak hanya dapat digunakan untuk melihat peringkat produktivitas, tetapi juga membantu memahami struktur kontribusi dalam ekosistem repositori open source.

Hasil sensitivity analysis menunjukkan bahwa DPI cukup stabil terhadap perubahan bobot. Korelasi Spearman yang sangat tinggi pada seluruh skenario mengindikasikan bahwa penggunaan equal-weight sebagai baseline masih dapat diterima. Namun, penelitian ini tetap menunjukkan bahwa pemilihan bobot dapat disesuaikan dengan kebutuhan analisis. Apabila penelitian ingin menekankan aktivitas kolaboratif, bobot PRP dapat ditingkatkan. Sebaliknya, apabila fokus analisis adalah volume perubahan kode, bobot CC dapat diperbesar.

Secara keseluruhan, hasil penelitian mengindikasikan bahwa Developer Productivity Index dapat menjadi pendekatan alternatif untuk mengevaluasi produktivitas kontributor pada proyek open source berbasis GitHub. DPI memberikan pengukuran yang lebih seimbang dibandingkan metrik tunggal karena menggabungkan intensitas aktivitas, konsistensi kontribusi, partisipasi kolaboratif, dan volume perubahan kode. Namun, DPI tetap memiliki keterbatasan karena belum mengukur kualitas kode, kompleksitas tugas, code review, issue resolution, dan konteks organisasi pengembangan. Oleh karena itu, pengembangan DPI pada penelitian selanjutnya dapat memasukkan metrik kualitas dan kolaborasi yang lebih luas agar hasil pengukuran menjadi lebih komprehensif.

4. Kesimpulan

Penelitian ini mengembangkan Developer Productivity Index (DPI) berbasis metrik repositori GitHub untuk mengevaluasi produktivitas kontributor pada proyek open source. DPI dibangun menggunakan empat metrik utama, yaitu Commit Frequency (CF), Active Days (AD), Pull Request Participation (PRP), dan Code Contribution (CC). Data penelitian diperoleh dari lima repositori open source, yaitu React, TensorFlow, Kubernetes, Visual Studio Code, dan Apache Spark selama periode 1 Januari 2024 sampai dengan 31 Desember 2024.

Hasil penelitian menunjukkan bahwa dari 1.560 akun kontributor yang dianalisis, terdapat 1.554 human contributor dan 6 bot account. Pemisahan akun bot dan human contributor menjadi penting karena beberapa akun otomatis memiliki aktivitas yang sangat tinggi dan dapat memengaruhi hasil pengukuran produktivitas. Pada hasil DPI seluruh akun, dua dari tiga akun dengan nilai DPI tertinggi merupakan akun bot. Setelah akun bot dipisahkan dan DPI human contributor dihitung ulang, hasil pemeringkatan menjadi lebih representatif untuk menggambarkan produktivitas developer manusia.

Hasil pengukuran DPI human contributor menunjukkan bahwa repositori VS Code memiliki rata-rata DPI tertinggi sebesar 0,0337. Akun Tyriar dari repositori VS Code memperoleh nilai DPI tertinggi sebesar 0,8693 karena memiliki kombinasi aktivitas yang kuat pada commit, active days, pull request, dan code contribution. Hasil validasi terhadap metrik tunggal menunjukkan bahwa DPI memberikan penilaian yang lebih seimbang karena tidak hanya bergantung pada satu indikator, tetapi menggabungkan beberapa dimensi aktivitas pengembangan perangkat lunak.

Analisis distribusi menunjukkan bahwa produktivitas kontributor pada proyek open source tidak terdistribusi secara merata. Nilai Gini yang tinggi pada metrik Commit Frequency, Pull Request Participation, Code Contribution, dan DPI mengindikasikan bahwa kontribusi cenderung didominasi oleh sejumlah kecil core contributor. Ketimpangan paling tinggi terdapat pada Code Contribution dengan nilai Gini sebesar 0,9636. Hasil sensitivity analysis juga menunjukkan bahwa pemeringkatan DPI relatif stabil terhadap perubahan bobot metrik, sehingga pendekatan equal-weight dapat digunakan sebagai baseline objektif dalam pengukuran produktivitas berbasis repositori GitHub.

Secara keseluruhan, DPI dapat menjadi pendekatan alternatif untuk mengevaluasi produktivitas kontributor pada proyek open source secara lebih komprehensif dibandingkan penggunaan metrik tunggal. Namun, penelitian ini masih memiliki keterbatasan karena belum mempertimbangkan aspek kualitas kode, kompleksitas tugas, aktivitas code review, issue resolution, serta konteks organisasi pengembangan. Penelitian selanjutnya dapat mengembangkan DPI dengan menambahkan metrik kualitas dan kolaborasi yang lebih luas agar hasil pengukuran produktivitas developer menjadi lebih akurat dan komprehensif.

5. UCAPAN TERIMA KASIH

Ucapan terima kasih kami sampaikan kepada Dosen Advance Software Engineering Program Studi Ilmu Komputer Universitas Nusa Mandiri dan teman-teman yang mendukung penelitian ini.

Referensi

[1] M. M. Ferreira, D. S. Goncalves, M. A. S. Bigonha, and K. A. M. Ferreira, *Characterizing Commits in Open-Source Software*, vol. 1, no. 1. Association for Computing Machinery, 2022. doi: 10.1145/3571473.3571508.

[2] M. Wessel, J. Vargovich, M. A. Gerosa, and C. Treude, "GitHub Actions: The Impact on the Pull Request Process," *Empir. Softw. Eng.*, vol. 28, no. 6, 2023, doi: 10.1007/s10664-023-10369-w.

[3] J. Santos, D. A. Da Costa, and U. Kulesza, *Investigating the Impact of Continuous Integration Practices on the Productivity and Quality of Open-Source Projects*, vol. 1, no. 1. Association for Computing Machinery, 2022. doi: 10.1145/3544902.3546244.

[4] K. Alrashedy and A. Binjahlan, "How do Software Engineering Researchers Use GitHub? An Empirical Study of Artifacts & Impact," *Proc. - 2024 IEEE Int. Conf. Source Code Anal. Manip. SCAM 2024*, pp. 118–130, 2024, doi: 10.1109/SCAM63643.2024.00021.

[5] Ł. Reszka, J. Sosnowski, and B. Dobrzyński, "Enhancing Software Project Monitoring with Multidimensional Data Repository Mining," *Electron.*, vol. 12, no. 18, 2023, doi: 10.3390/electronics12183774.

[6] A. Noda, M. A. Storey, N. Forsgren, and M. Greiler, "DevEX: What Actually Drives Productivity?," *Commun. ACM*, vol. 66, no. 11, pp. 44–49, 2023, doi: 10.1145/3610285.

[7] N. Forsgren, E. Kalliamvakou, A. Noda, M. Greiler, B. Houck, and M. A. Storey, "DevEx in Action A study of its tangible impacts," *Queue*, vol. 21, no. 6, pp. 47–77, 2023, doi: 10.1145/3639443.

[8] M. A. Storey, B. Houck, and T. Zimmermann, *How Developers and Managers Define and Trade Productivity for Quality*, vol. 1, no. 1. Association for Computing Machinery, 2022. doi: 10.1145/3528579.3529177.

[9] Y. Sun *et al.*, "Automatically Deriving Developers' Technical Expertise from the GitHub Social Network," *ACM Trans. Softw. Eng. Methodol.*, vol. 35, no. 4, 2026, doi: 10.1145/3746451.

[10] J. T. Liang, T. Zimmermann, and D. Ford, *Towards mining OSS skills from GitHub activity*, vol. 1, no. 1. Association for Computing Machinery, 2022. doi: 10.1145/3510455.3512772.

[11] Q. Perez, C. Urtado, and S. Vauttier, "Dataset of open-source software developers labeled by their experience

level in the project and their associated software metrics," *Data Br.*, vol. 46, p. 108842, 2023, doi: 10.1016/j.dib.2022.108842.

[12] I. S. Göçmen, A. S. Cezayir, and E. Tüzün, *Enhanced code reviews using pull request based change impact analysis*, vol. 30, no. 3. 2025. doi: 10.1007/s10664-024-10600-2.

[13] G. Kudrjavets and A. Rastogi, *Does code review speed matter for practitioners?*, vol. 29, no. 1. 2024. doi: 10.1007/s10664-023-10401-z.

[14] A. Trautsch, J. Erbel, S. Herbold, and J. Grabowski, "What really changes when developers intend to improve their source code: a commit-level study of static metric value and static analysis warning changes," *Empir. Softw. Eng.*, vol. 28, no. 2, 2023, doi: 10.1007/s10664-022-10257-9.

[15] A. K. Turzo and A. Bosu, "What makes a code review useful to OpenDev developers? An empirical investigation," *Empir. Softw. Eng.*, vol. 29, no. 1, 2024, doi: 10.1007/s10664-023-10411-x.

[16] K. A. Hasan, M. MacEdo, Y. Tian, B. Adams, and S. Ding, "Understanding the Time to First Response in GitHub Pull Requests," *Proc. - 2023 IEEE/ACM 20th Int. Conf. Min. Softw. Repos. MSR 2023*, pp. 1–11, 2023, doi: 10.1109/MSR59073.2023.00015.

[17] Y. Sun, Z. Xu, C. Liu, Y. Zhang, and Y. Liu, "Who is the Real Hero? Measuring Developer Contribution via Multi-Dimensional Data Integration," *Proc. - 2023 38th IEEE/ACM Int. Conf. Autom. Softw. Eng. ASE 2023*, pp. 825–836, 2023, doi: 10.1109/ASE56229.2023.00102.

[18] P. Thongtanunam and C. Tantithamthavorn, "Code Ownership: The Principles, Differences, and Their Associations with Software Quality," *Proc. - Int. Symp. Softw. Reliab. Eng. ISSRE*, pp. 379–390, 2024, doi: 10.1109/ISSRE62328.2024.00044.

[19] D. S. M. Varanasi, D. Divij, Y. Raghu Reddy, and S. A. Karre, "Measuring Software Development Waste in Open-Source Software Projects," *Proc. Euromicro Conf. Softw. Eng. Adv. Appl. EUROMICRO-SEAA*, no. 2024, pp. 276–283, 2024, doi: 10.1109/SEAA64295.2024.00050.

[20] U. Shakikhanli and V. Bilicki, "Repository structures: Impact on collaboration and productivity," *Pollack Period.*, vol. 20, no. 1, pp. 33–38, 2025, doi: 10.1556/606.2024.01123.

[21] A. Foundjem, E. Constantinou, T. Mens, and B. Adams, "A mixed-methods analysis of micro-collaborative coding practices in OpenStack," *Empir. Softw. Eng.*, vol. 27, no. 5, 2022, doi: 10.1007/s10664-022-10167-w.

[22] S. Peng, E. Kalliamvakou, P. Cihon, and M. Demirer, "The Impact of AI on Developer Productivity: Evidence from GitHub Copilot," pp. 1–19, 2023, doi: https://doi.org/10.48550/arXiv.2302.06590.

[23] V. Stray, E. G. Brandtzæg, V. T. Wivestad, A. Barbala, and N. B. Moe, "Developer Productivity With and Without GitHub Copilot: A Longitudinal Mixed-Methods Case Study," *59th Hawaii Int. Conf. Syst. Sci.*, 2026, doi: https://doi.org/10.48550/arXiv.2509.20353.

[24] A. Ziegler *et al.*, "Measuring github copilot's impact on productivity," *Commun. ACM*, vol. 67, no. 3, pp. 54–63, 2024, doi: 10.1145/3633453.

[25] F. Song, A. Agarwal, and W. Wen, "The Impact of Generative AI on Collaborative Open-Source Software Development: Evidence from GitHub Copilot," *SSRN Electron. J.*, 2024, doi: 10.2139/ssrn.4856935.

[26] V. Riyanto, H. Destiana, T. Prihatin, Sugiono, and G. Wijaya, "Mengoptimalkan Prediksi Gagal Jantung Dengan Kombinasi Svm Dan Forward Selection," *J. Inform. dan Rekayasa Elektron.*, vol. 8, no. 1, pp. 103–111, 2025, doi: 10.36595/jire.v8i1.1541.

[27] R. Yuranda *et al.*, "Deteksi aktivitas mencurigakan pada log nginx menggunakan isolation forest berbasis analisis perilaku," vol. 9, no. 1, pp. 74–83, 2026.