Tugas Akhir: Lapor Riset Empirik IMK

Platform Low-Code untuk Meningkatkan Interaksi *Software Engineer*

Nur Rahmat Dwi Riyanto – 6025222008

A. Peta Pikir Makalah

https://miro.com/app/board/uXjVNFFIURo=/?share_link_id=163867669621

B. Manuskrip Bahasa Indonesia

Pendahuluan

Tim pengembangan merasa tertekan karena harus tanggap terhadap meningkatnya harapan pelanggan dan meningkatkan kecepatan dalam melakukan transformasi digital. Selain itu, permintaan untuk inovasi solusi bisnis baru diprediksi akan tetap tinggi dalam waktu yang akan dating [11].

Metode pengembangan konvensional tidak dirancang untuk mengakomodasi tuntutan kecepatan yang ada saat ini, dan pertumbuhan pesat adopsi SaaS (Perangkat Lunak sebagai Layanan) yang dilakukan oleh tim bisnis telah mencapai tingkat yang menghambat produktivitas. Selain itu, merekrut talenta pengembang saat ini menjadi lebih sulit, di mana jumlah insinyur tidak mencukupi untuk memenuhi kebutuhan pembangunan yang sedang berlangsung [11].

Platform low-code merupakan pendekatan dalam pengembangan perangkat lunak yang memberikan kemampuan kepada pengembang untuk membuat aplikasi dengan coding yang minimal. Dengan menggunakan low-code, pengembang dapat menggunakan bahasa visual berbasis model dan antarmuka grafis drag-and-drop [11].

Dengan keunggulan pengembangan visual dan otomatisasi yang dimilikinya, platform *low-code* memungkinkan pengembang untuk membuat aplikasi yang lengkap, termasuk antarmuka pengguna modern, integrasi, data, dan logika, dengan waktu yang jauh lebih singkat dibandingkan dengan pengembangan tradisional [11]. Dalam kategori yang dikenal sebagai *'low-code'*, serangkaian lingkungan pengembangan perangkat lunak baru telah muncul dalam beberapa tahun terakhir.

Lingkungan ini tidak hanya menjanjikan peningkatan yang signifikan dalam produktivitas pengembangan perangkat lunak namun juga menawarkan pendekatan baru untuk mendorong keselarasan antara IT *bussiness* dan tim pengembang. Istilah yang digunakan untuk menggambarkan platform ini mencakup *low-code platform* (LCP), *low-code application platform* (LCAP), and *low-code development platform* (LCDP) [1].

Tinjauan Pustaka

Survei teknis tentang berbagai platform low-code dan studi perbandingannya telah disajikan [8]. Studi ini memberikan gambaran umum dan analisis berbagai platform low-code, termasuk Kissflow, platform berbasis cloud yang berfokus pada otomatisasi alur kerja untuk proses bisnis. Studi lain [3] mengeksplorasi pengalaman pengembang dari platform *low code* dan memberikan rekomendasi untuk meningkatkan pengalaman pengembang. Selain itu, sebuah penelitian [11] memberikan penjelasan yang seimbang tentang tren pengembangan low code saat ini dan menempatkan topik dalam konteks penelitian sistem informasi bisnis yang lebih luas. Penelitian ini membahas pertanyaan-pertanyaan seperti ciri-ciri karakteristik platform low-code, bagaimana perbandingannya dengan status penelitian saat ini, dan inovasi teknologi apa yang diwujudkan oleh platform-platform ini.

Penelitian yang dilakukan [2] memperkenalkan kerangka kerja interaksi manusia-LLM baru yang menggabungkan enam jenis interaksi pemrograman visual kode rendah sederhana untuk mencapai respons yang lebih terkendali dan stabil. Kerangka kerja LLM kode rendah yang diusulkan terdiri dari LLM Perencanaan yang merancang alur kerja perencanaan terstruktur untuk tugas-tugas kompleks, yang dapat diedit dan dikonfirmasi oleh pengguna melalui operasi pemrograman visual kode rendah, dan LLM Pelaksana yang menghasilkan respons mengikuti perilaku pengguna. alur kerja yang dikonfirmasi. Makalah ini menyoroti tiga keunggulan LLM berkode rendah: hasil pembangkitan yang dapat dikontrol, interaksi manusia-LLM yang mudah digunakan, dan skenario yang dapat diterapkan secara luas. Studi [12] membahas bagaimana alat tanpa kode atau kode rendah sudah dapat mendukung alur pengembangan aplikasi ML yang biasanya memerlukan pengkodean. Terakhir, artikel [9] menjelaskan bagaimana para peneliti Microsoft menerbitkan makalah tentang model bahasa besar (LLM) berkode rendah yang dapat digunakan untuk proyek pembelajaran mesin seperti ChatGPT, chatbot yang terdengar seperti makhluk hidup dari OpenAI.

Metodologi

Metodologi pada penelitian ini di dasarkan pada metode NASA Task Load Index (TLX). Metode NASA-TLX digunakan untuk evaluasi beban kerja mental pada individu yang terlibat dalam berbagai kegiatan pekerjaan. Dikembangkan pada tahun 1981 oleh Sandra G. Hart dari NASA-Ames Research Center dan Lowell E. Staveland dari San Jose State University, metode ini muncul sebagai respons terhadap kebutuhan untuk mengukur secara subjektif faktor-faktor seperti kesulitan tugas, tekanan waktu, jenis aktivitas, usaha fisik, usaha mental, performansi, frustasi, stres, dan kelelahan, yang kemudian disederhanakan menjadi enam faktor: *Mental demand* (MD), *Physical demand* (PD), *Temporal demand* (TD), *Own Performance* (PO), *Effort* (E), *Frustation level* (FR).

NASA-TLX merupakan metode subjektif untuk mengukur beban kerja mental. Proses pengukuran metode NASA-TLX terdiri dari dua tahap, yakni perbandingan antar skala (Paired Comparison) dan penilaian terhadap pekerjaan (Event Scoring) [8].

Aparatus

Aparatus yang di gunakan pada penelitian ini terdiri dari 2 framework sebagai bahan perbandingan, yaitu :

1. **React JS** : Sebuah pustaka (library) JavaScript yang digunakan untuk membangun antarmuka pengguna (UI) dalam aplikasi web. React dikembangkan oleh Facebook dan dirilis sebagai proyek open source. Salah satu fitur utama React adalah kemampuannya untuk membuat tampilan UI yang responsif dan efisien dengan menggunakan konsep komponen.

2. **OutSystems** : Sebuah platform pengembangan aplikasi berbasis low-code yang memungkinkan pengembang untuk membuat, menguji, dan menyebarkan aplikasi perangkat lunak dengan cepat dan efisien. Pendekatan low-code dalam pengembangan aplikasi berarti bahwa pengguna dapat membuat aplikasi dengan lebih sedikit kode manual, mengurangi kompleksitas pengembangan dan mempercepat waktu rilis aplikasi.

Partisipan

Partisipan pada penelitian ini terdiri dari 6 developer sistem profesional dengan latar belakang dan pengalaman yang berbeda. Partisipan di minta untuk mengerjakan tugas-tugas yang telah di berikan melalui suatu tutorial yang telah di berikan.

Skenario Tugas

Para pengembang diminta untuk menyelesaikan tugas pengembangan sistem yang terdiri dari tiga bagian, yaitu:

1. **Pre - Task** : Melakukan penyiapan perangkat lunak pendukung pada link https://bit.ly/tugas-imk-master dan membuat project React JS baru, dan/atau melakukan login ke dalam platform OutSystems.

   - Pre-Task React JS.
   - Pre-Task OutSystems.

   Untuk melakukan login ke platform OutSystems partisipan dapat menggunakan akun di bawah ini:

   Email : tugasimkrahmat@gmail.com
   Password : tugas!MK1234

2. **Task - 1** : Membangun aplikasi dengan tujuan menampilkan halaman sederhana bertuliskan "Hello World."

- Task-1 React JS.
- Task-1 OutSystems.

3. **Task - 2** : Membangun aplikasi dengan fokus menampilkan halaman yang berisi daftar siswa.

- Task-2 React JS.
- Task-2 OutSystems.

Setelah partisipan selesai mengerjakan tugas-tugas di atas. Partisipan di minta untuk mengisi formulir survey berdasarkan NASA Task Load Index (TLX) yaitu dengan kategori penilaian per indikator adalah sebagai berikut:

Tabel 4.3 Skor NASA-TLX

Golongan Beban Kerja	Nilai
Rendah	0 - 9
Sedang	10 - 29
Agak Tinggi	30 - 49
Tinggi	50 - 79
Sangat Tinggi	80 - 100

Gambar 1. Interpretasi Skor NASA TLX

Langkah Pengukuran NASA-TLX

Langkah-langkah pengukuran dengan menggunakan NASA TLX adalah sebagai berikut [4]:

1. **Pembobotan:** Dalam segmen ini, peserta diminta memilih di antara dua indikator yang dianggap lebih mendominasi dalam menciptakan beban kerja mental terkait pekerjaan tersebut. Kuesioner NASA-TLX yang diberikan berbentuk perbandingan pasangan. Dari kuesioner ini, dihitung total jumlah tanda (tally) untuk setiap indikator yang dianggap memiliki dampak paling signifikan. Total tanda kemudian menjadi bobot untuk masing-masing indikator beban kerja mental.

2. **Pemberian Rating:** Pada segmen ini, peserta diminta memberikan penilaian terhadap enam indikator beban mental. Penilaian yang diberikan bersifat subjektif dan bergantung pada persepsi beban mental yang dirasakan oleh masing-masing

responden. Untuk menghitung skor beban mental NASATLX, bobot dan penilaian untuk setiap indikator dikalikan, kemudian hasilnya dijumlahkan dan dibagi dengan 15 (jumlah perbandingan berpasangan).

3. **Menghitung Nilai Produk:** Diperoleh dengan melakukan perkalian antara penilaian dan bobot faktor untuk setiap deskriptor. Oleh karena itu, dihasilkan enam nilai produk yang mewakili enam indikator (MD, PD, TD, CE, FR, EF):

$$Produk = rating \times bobot\ faktor$$

Persamaan 1. Perhitungan nilai produk

4. **Menghitung *Weighted Workload* (WWL):** Menghitung dengan menjumlahkan keenam nilai produk

$$WWL = \sum produk$$

Persamaan 2. Perhitungan WWL

5. **Menghitung rata-rata WWL:** Diperoleh dengan membagi WWL dengan jumlah bobot total:

$$Skor = \frac{\sum produk}{15}$$

Persamaan 3. Rata-rata WWL

6. **Interpretasi Skor:** Interpretasi dilakukan untuk mengetahui tingkat beban kerja peserta sesuai pada Gambar 1.

Hasil dan Diskusi

Setelah para developer selesai melakukan tugas-tugas yang telah di berikan, dan mengisi kuisioner sesuai indikator pada Gambar 1. Hasil data dari kuisioner para partisipan kemudian dilakukan pengolahan lebih lanjut sesuai dengan tahapan-tahapan pada metode NASA TLX.

1. Pembobotan

Peneliti melakukan pembobotan pada setiap 6 indikator beban kerja partisipan, dengan nilai bobot sebagai berikut:

Tabel 1. Tabel Pembobotan Indikator

Data Pembobotan Kuisioneer						
Indikator						
MD	**PD**	**TD**	**OP**	**E**	**FR**	**Total**
2	2	2	2	4	3	15

Total dari pembobotan masing-masing indikator adalah 15. Tabel 1 menunjukkan bahwa tuntutan mental (MD), tuntutan fisik (PD), tuntutan beban waktu (TD), kinerja (OP),

usaha (E), dan frustrasi(FR) adalah faktor-faktor yang dinilai. Indikator usaha (E) adalah indikator yang memiliki nilai paling besar di bandingkan yang lain yaitu bernilai 4, karena menurut peneliti indikator ini adalah indikator yang memiliki pengaruh signifikan dalam proses interaksi antara developer dan platform.

2. Pemberian Rating

Peneliti melakukan proses pengumbulan data hasil kuisioner, dan membagi menjadi 2 data terpisah berdasarkan kategori apparatus. Hasil dari data pemberian rating tersebut dapat di lihat pada Tabel 2.

Tabel 2. Tabel Hasil Rating OutSystems dan React JS

Data Hasil Rating OutSystems							
Objek Penelitian		**Indikator**					
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**
Participant 1	4	10	15	10	50	10	10
Participant 2	3	25	5	30	25	10	8
Participant 3	3	30	30	90	70	30	20
Participant 4	5	20	25	10	60	30	10
Participant 5	2	100	65	100	69	79	15
Participant 6	3	50	60	70	80	70	50
Data Hasil Rating React JS							
Objek Penelitian		**Indikator**					
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**
Participant 1	4	10	15	10	50	10	10
Participant 2	3	15	5	20	20	15	5
Participant 3	3	50	70	70	20	70	30
Participant 4	5	50	50	60	30	70	90
Participant 5	2	100	65	100	65	70	20
Participant 6	3	60	70	80	80	80	60

3. Perhitungan Nilai Produk

Pada tahap ini peneliti melakukan perhitungan nilai produk sesuai dengan Persamaan 1. Hasil nilai produk pada masing-masing partisipan dapat di lihat pada Table 3 dan Tabel 4.

Tabel 3. Hasil Rating OutSystems

Total Nilai Produk OutSystems							
Objek Penelitian		**Indikator**					
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**
Participant 1	4	20	30	20	100	40	30
Participant 2	3	50	10	60	50	40	24
Participant 3	3	60	60	180	140	120	60
Participant 4	5	40	50	20	120	120	30
Participant 5	2	200	130	200	138	316	45
Participant 6	3	100	120	140	160	280	150

Tabel 4. Hasil Rating React JS

Total Nilai Produk React JS							
Objek Penelitian		Indikator					
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**
Participant 1	4	20	30	20	100	40	30
Participant 2	3	30	10	40	40	60	15
Participant 3	3	100	140	140	40	280	90
Participant 4	5	100	100	120	60	280	270
Participant 5	2	200	130	200	130	280	60
Participant 6	3	120	140	160	160	320	180

4. Perhitungan WWL (Weighted Workload)

Perhitungan weighted workload di lakukan sesuai dengan persamaan 2, yaitu dengan menjumlah total nilai produk masing-masing indikator setiap partisipan. Hasil dari penjumlahan tersebut dapat di lihat pada Tabel 5.

Tabel 5. WWL OutSystems dan ReactJS

Total Nilai Produk OutSystems								Total Nilai Weighted Workload OutSystems
Objek Penelitian		Indikator						
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**	**Total**
Participant 1	4	20	30	20	100	40	30	240
Participant 2	3	50	10	60	50	40	24	234
Participant 3	3	60	60	180	140	120	60	620
Participant 4	5	40	50	20	120	120	30	380
Participant 5	2	200	130	200	138	316	45	1029
Participant 6	3	100	120	140	160	280	150	950

Total Nilai Produk React JS								Total Nilai Weighted Workload React JS
Objek Penelitian		Indikator						
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**	**Total**
Participant 1	4	20	30	20	100	40	30	240
Participant 2	3	30	10	40	40	60	15	195
Participant 3	3	100	140	140	40	280	90	790
Participant 4	5	100	100	120	60	280	270	930
Participant 5	2	200	130	200	130	280	60	1000
Participant 6	3	120	140	160	160	320	180	1080

5. Rata-Rata WWL (Weighted Workload)

Setelah mendapatkan nilai WWL pada setiap partisipan. Langkah selanjutnya peneliti melakukan rata-rata nilai WWL sesuai dengan Persamaan 3 dimana setiap total WWL masing-masing partisipan di bagi dengan total indikator yaitu 15. Hasil perhitungan rata-rata WWL dapat di lihat pada Tabel 6.

Tabel 6. Rata-rata WWL OutSystems dan ReactJS

Perhitungan Rata-rata Weighted Workload (WWL) OutSystems								
Objek Penelitian		Indikator						
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**	**Total**
Participant 1	4	1.333333333	2	1.333333333	6.666666667	2.666666667	2	16
Participant 2	3	3.333333333	0.666666667	4	3.333333333	2.666666667	1.6	15.6
Participant 3	**3**	**4**	**4**	**12**	**9.333333333**	**8**	**4**	**41.33333333**
Participant 4	**5**	**2.666666667**	**3.333333333**	**1.333333333**	**8**	**8**	**2**	**25.33333333**
Participant 5	2	13.33333333	8.666666667	13.33333333	9.2	21.06666667	3	68.6
Participant 6	**3**	**6.666666667**	**8**	**9.333333333**	**10.66666667**	**18.66666667**	**10**	**63.33333333**

Perhitungan Rata-rata Weighted Workload (WWL)								
Objek Penelitian		Indikator						
Nama	**Pengalaman**	**MD**	**PD**	**TD**	**OP**	**E**	**FR**	**Total**
Participant 1	4	1.333333333	2	1.333333333	6.666666667	2.666666667	2	16
Participant 2	3	2	0.666666667	2.666666667	2.666666667	4	1	13
Participant 3	3	6.666666667	9.333333333	9.333333333	2.666666667	18.66666667	6	52.66666667
Participant 4	5	6.666666667	6.666666667	8	4	18.66666667	18	62
Participant 5	2	13.33333333	8.666666667	13.33333333	8.666666667	18.66666667	4	66.66666667
Participant 6	3	8	9.333333333	10.66666667	10.66666667	21.33333333	12	72

6. Interpretasi Skor

Tahap akhir pada penelitian adalah melakukan interprestasi skor yang di peroleh dari Rata-Rata WWL masing-masing partisipan ketika mengerjakan tugas menggunakan OutSystems maupun React JS. Interpretasi ini di didasarkan pada skor pada Gambar 1. Hasil interpretasi tersebut di sajikan pada Tabel 7 dan Tabel 8.

Tabel 7. Interprestasi Beban Kerja React JS

Kategori Penilaian Beban Kerja React JS		
Nama	Nilai Beban Kerja	Kategori
Participant 1	16	Sedang
Participant 2	13	Sedang
Participant 3	52.66666667	Tinggi
Participant 4	62	Tinggi
Participant 5	66.66666667	Tinggi
Participant 6	72	Tinggi

Dalam konteks interpretasi, dapat disimpulkan bahwa Beban Kerja pada framework React JS menunjukkan rata-rata yang berkategori skor tinggi pada partisipan 3 hingga partisipan 6. Sementara itu, partisipan 1 dan 2 menunjukkan beban kerja dengan kategori skor sedang.

Tabel 8. Interprestasi Beban Kerja OutSystems

Kategori Penilaian Beban Kerja OutSystems		
Nama	Nilai Beban Kerja	Kategori
Participant 1	16	Sedang
Participant 2	15.6	Sedang
Participant 3	41.33333333	**Agak Tinggi**
Participant 4	25.33333333	**Sedang**
Participant 5	68.6	Tinggi
Participant 6	63.33333333	Tinggi

Dalam analisis interpretatif, dapat disimpulkan bahwa Beban Kerja pada platform OutSystems menunjukkan rata-rata yang berkategori skor sedang pada partisipan 1, partisipan 2, dan partisipan 4. Sebaliknya, partisipan 5 dan 6 menunjukkan tingkat beban kerja yang berkategori skor tinggi. Di sisi lain, partisipan 3 menunjukkan skor yang berkategori agak tinggi dalam hal beban kerja.

Kesimpulan dan Saran

Kesimpulan

Dari hasil penilaian beban kerja pada platform OutSystem dan React JS terhadap enam partisipan, dapat disimpulkan bahwa partisipan 3 menunjukkan kategori beban kerja yang agak tinggi pada kedua platform, dengan nilai masing-masing sebesar 41,33 untuk OutSystem dan 52,67 untuk React JS. Partisipan 5 dan 6 menunjukkan kategori beban kerja yang tinggi pada kedua platform, dengan nilai tertinggi pada React JS sebesar 72. Sementara itu, partisipan 1 dan 2 menunjukkan kategori beban kerja yang berada pada tingkat sedang, dengan nilai yang bervariasi antara 13 hingga 16. Dengan demikian, dapat ditarik kesimpulan bahwa React JS cenderung menunjukkan kategori beban kerja yang lebih tinggi dibandingkan OutSystem, dan variasi dalam penilaian beban kerja antar partisipan dapat diidentifikasi melalui nilai-nilai yang diberikan.

Saran

Beberapa saran yang diperlukan dalam penelitian selanjutnya adalah sebagai berikut:

- Penambahan skenario yang lebih kompleks guna mendapatkan pemahaman yang lebih mendalam terkait beban partisipan.
- Penambahan partisipan dalam tahap pengujian guna mendapatkan hasil data yang lebih bervariasi.
- Menambah low code platform yang di uji untuk mengetahui lebih luas mengenai perspektif penelitian ini terhadap Low Code Platform.

Daftar Pustaka

[1] Bock, A. C., & Frank, U. (2021a). Low-Code Platform. *Business & Information Systems Engineering*, *63*(6), 733–740. https://doi.org/10.1007/s12599-021-00726-8

[2] Cai, Y., Mao, S., Wu, W., Wang, Z., Liang, Y., Ge, T., Wu, C., You, W., Song, T., Xia, Y., Tien, J., & Duan, N. (2023). *Low-code LLM: Visual Programming over LLMs* (arXiv:2304.08103). arXiv. https://doi.org/10.48550/arXiv.2304.08103

[3] Dahlberg, D. (n.d.). *Developer Experience of a Low-Code Platform: An exploratory study*.

[4] Hancock, P.A., dan N. Meshkati. (1988). Human Mental Workload. Los Angeles: University of Southern California.

[5] Juhas, G., Molnar, L., Juhasova, A., Ondrisova, M., Mladoniczky, M., & Kovacik, T. (2022). Low-code platforms and languages: The future of software development. *2022 20th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 286–293. https://doi.org/10.1109/ICETA57911.2022.9974697

[6] Kim, T. S., Choi, D., Choi, Y., & Kim, J. (2022). Stylette: Styling the Web with Natural Language. *CHI Conference on Human Factors in Computing Systems*, 1–17. https://doi.org/10.1145/3491102.3501931

[7] Martins, J., Branco, F., & Mamede, H. (2023). Combining low-code development with ChatGPT to novel no-code approaches: A focus-group study. *Intelligent Systems with Applications*, *20*, 200289. https://doi.org/10.1016/j.iswa.2023.200289

[8] Pradhana, C. A., & Mt, H. S. S. (n.d.). *ANALISIS BEBAN KERJA MENTAL MENGGUNAKAN METODE NASA-TLX PADA BAGIAN SHIPPING PERLENGKAPAN DI PT. TRIANGLE MOTORINDO*.

[9] Ramel, B. D., & 04/26/2023. (n.d.). *Microsoft Researchers Tackle Low-Code LLMs -*. Visual Studio Magazine. Retrieved December 10, 2023, from https://visualstudiomagazine.com/articles/2023/04/26/low-code-llms.aspx

[10] Talesra, K., & G. S., N. (2021). Low-Code Platform for Application Development. *International Journal of Applied Engineering Research*, *16*(5), 346. https://doi.org/10.37622/IJAER/16.5.2021.346-351

[11] *What is Low-Code | Low-Code Guide*. (n.d.). Retrieved December 9, 2023, from https://www.outsystems.com/guide/low-code/

[12] *How Can No/Low Code Platforms Help End-Users Develop ML Applications? - A Systematic Review | HCI International 2022 – Late Breaking Papers: Interacting with eXtended Reality and Artificial Intelligence*. (n.d.). Retrieved December 10, 2023, from https://dl.acm.org/doi/abs/10.1007/978-3-031-21707-4_25