BAB VI
KESIMPULAN DAN SARAN

6.1. Kesimpulan

Tujuan pertama penelitian tercapai melalui terbangunnya platform *coding contest* terdistribusi yang tersusun atas 10 layanan *microservices* berbasis Java 21 dan Spring Boot. Komunikasi asinkron diwujudkan melalui RabbitMQ yang memisahkan penerimaan submisi dari proses penilaian, sehingga *blocking process* pada jalur permintaan tereliminasi. Mekanisme *warm worker pool* mempertahankan kontainer *judge worker* tetap aktif, sehingga setiap submisi langsung dieksekusi tanpa jeda *cold-start* pembuatan kontainer baru. Eksekusi kode peserta berlangsung terisolasi di dalam *sandbox* Isolate pada kontainer Docker dengan kuota sumber daya yang dikunci melalui *cgroups*. Seluruh 28 butir uji fungsional dinyatakan handal dengan tingkat keberhasilan 100 persen, sehingga keseluruhan fitur platform beroperasi sesuai spesifikasi.

Tujuan kedua penelitian tercapai melalui pembuktian setelah evaluasi keunggulan kinerja atas *online judge* konvensional pada pengujian beban hingga 1.000 pengguna virtual di lingkungan lokal dan cloud. Arsitektur *microservices* mempertahankan *error rate* 0 persen dengan *throughput* 2,7 sampai 3,7 kali lebih besar serta latensi yang selalu lebih rendah, sementara *monolith* gagal melayani hingga 84,3 persen permintaan pada beban puncak. Keadilan penilaian terbukti melalui koefisien variasi waktu eksekusi di bawah ambang 1 persen dengan seluruh 300 eksekusi menghasilkan *verdict* identik. Keseluruhan hasil tersebut menjawab rumusan masalah mengenai *blocking process*, skalabilitas, dan inkonsistensi waktu penilaian, sehingga lonjakan submisi tidak lagi menggugurkan solusi peserta dan setiap peserta dinilai dengan pengukuran waktu yang setara. Kontribusi nyata penelitian berupa bukti empiris keunggulan arsitektur tersebut beserta rancangan sistem yang dapat menjadi acuan pembangunan *online judge* terdistribusi yang andal dan adil.

6.2. Saran

Saran bagi pengembang yang melanjutkan penelitian ini diarahkan pada tiga hal yang bersumber dari keterbatasan sistem. Pertama, penskalaan *judge worker* perlu diotomasi melalui orkestrasi Kubernetes dengan panjang antrean penilaian sebagai dasar keputusan, sehingga kapasitas mengikuti beban tanpa langkah manual. Kedua, infrastruktur penunjang berupa RabbitMQ, PostgreSQL, dan MinIO perlu direplikasi agar kegagalan satu komponen tidak menghentikan keseluruhan alur penilaian. Ketiga, pengujian lanjutan perlu mencakup beban yang berlangsung lama, tingkat konkurensi di atas 1.000 pengguna virtual, serta kode dengan beban memori dan masukan keluaran pada pengujian konsistensi.

Saran bagi penyelenggara kompetisi sebagai pengguna platform berfokus pada pengelolaan kapasitas alokasi *worker* manual. Jumlah kontainer *judge worker* harus disiapkan sebelum kontes berdasarkan perkiraan jumlah peserta, kemudian panjang antrean penilaian perlu dipantau melalui halaman pemantauan selama kontes berlangsung. Institusi pendidikan juga dapat memanfaatkan platform ini untuk latihan dan ujian pemrograman, karena konsistensi penilaian dan kapasitas yang telah terbukti sesuai dengan kebutuhan kelas pemrograman berukuran besar.

DAFTAR PUSTAKA

[1] H. Z. Dosilovic and I. Mekterovic, "Robust and Scalable Online Code Execution System," in *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, IEEE, Sep. 2020, pp. 1627–1632. doi: 10.23919/MIPRO48935.2020.9245310.

[2] Crişan, Daniela Alexandra Stănică, Justina Lavinia Altăr-Samuel, and Adam Nelu, "Competitive Programming: An Analysis of the Performance in Romania," *Journal of Information Systems & Operations Management*, vol. 14, no. 1, pp. 43–59, 2020.

[3] Y. SU, P. Nie, and X. MENG, "OI-Assistant: A Retrieval Augmented System for Similar Problem Discovery and Interactive Learning in Competitive Programming," *OLYMPIADS IN INFORMATICS*, pp. 129–144, 2025, doi: 10.15388/ioi.2025.09.

[4] G. AUDRITO, L. LAURA, A. ORLANDI, D. OSTUNI, R. RIZZI, and L. VERSARI, "Interactive Problem Solving in the Classroom: Experiences with Turing Arena Light in Competitive Programming Education," *OLYMPIADS IN INFORMATICS*, pp. 1–25, 2025, doi: 10.15388/ioi.2025.01.

[5] J. Liang and C. Tu, "Design and implementation of online judgment system based on microservice architecture and Docker," in *2022 5th International Conference on Advanced Electronic Materials, Computers and Software Engineering (AEMCSE)*, IEEE, Apr. 2022, pp. 749–753. doi: 10.1109/AEMCSE55572.2022.00150.

[6] G.-C. Pan, P. Liu, and J.-J. Wu, "A Cloud-Native Online Judge System," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, IEEE, Jun. 2022, pp. 1293–1298. doi: 10.1109/COMPSAC54236.2022.00204.

[7] Kamod, Nachiket Devendra Jadhav, and Rahul Namdev, "A Secure and Scalable System for Online Code Execution and Evaluation using

Containerization and Kubernetes," *Journal of Emerging Technologies and Innovative Research (JETIR)*, vol. 10, no. 2, pp. c151–c159, 2023.

[8] P. DIETRICH and B. KOSTKA, "Virtual Time Measurement in Programming Contests," *OLYMPIADS IN INFORMATICS*, pp. 27–33, 2025, doi: 10.15388/ioi.2025.02.

[9] G. Pan, P. Liu, and J.-J. Wu, "A Cloud-Native Online Judge System," *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2022.

[10] V. Velepucha and P. Flores, "A Survey on Microservices Architecture: Principles, Patterns and Migration Challenges," *IEEE Access*, vol. 11, pp. 88339–88358, 2023, doi: 10.1109/ACCESS.2023.3305687.

[11] Z. Yu, J. Han, T. Zhao, N. Tian, and J. Wang, "Research and Implementation of Online Judgment System Based on Micro Service," in *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, IEEE, Oct. 2019, pp. 475–478. doi: 10.1109/ICSESS47205.2019.9040684.

[12] H. Yibo, Z. Zhang, B. Yuan, H. Bi, M. N. Shahzad, and L. Liu, "An Experimental Online Judge System Based on Docker Container for Learning and Teaching Assistance," in *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, IEEE, Aug. 2019, pp. 1462–1467. doi: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00264.

[13] A. Ariff, "Rancang Bangun Aplikasi Online Judge Untuk Aplikasi Berbasis Jaringan Memanfaatkan Virtualisasi Berbasis Docker," Institut Teknologi Sepuluh Nopember, Surabaya, 2015.

[14] Y. Watanobe, Md. M. Rahman, T. Matsumoto, U. K. Rage, and P. Ravikumar, "Online Judge System: Requirements, Architecture, and

Experiences," *International Journal of Software Engineering and Knowledge Engineering*, vol. 32, no. 06, pp. 917–946, Jun. 2022, doi: 10.1142/S0218194022500346.

[15] A. Arvon, M. Muthahhari, and A. Perwitasari, "Pengembangan Microservice Grader pada Online Judge Berbasis Event-Driven Architecture," *JEPIN (Jurnal Edukasi dan Penelitian Informatika)*, vol. 10, no. 3, pp. 446–459, 2024.

[16] A. LAAKSONEN and T. TALVITIE, "CSES – Yet Another Online Judge," *OLYMPIADS IN INFORMATICS*, pp. 105–111, Dec. 2020, doi: 10.15388/ioi.2020.08.

[17] F. HERNÁNDEZ GONZÁLEZ, J. D. RODRÍGUEZ MORALES, and D. A. RIPOLL MÉNDEZ, "The Cuban Olympiad in Informatics: A New Stage from the DMOJ Online Judge," *OLYMPIADS IN INFORMATICS*, pp. 133–141, Jun. 2021, doi: 10.15388/ioi.2021.11.

[18] D. TSEDEVSUREN, J. DASHDEMBEREL, T.-O. BATTOGTOKH, T. ULAMBAYAR, and A. KHUDER, "Organization and Results of Mongolian National Online Olympiads in Informatics," *OLYMPIADS IN INFORMATICS*, pp. 145–158, 2022, doi: 10.15388/ioi.2022.12.

[19] J.-C. Liang, J.-Y. Wang, and S.-H. Chou, "A designed platform for programming courses," in *2020 9th International Congress on Advanced Applied Informatics (IIAI-AAI)*, IEEE, Sep. 2020, pp. 290–293. doi: 10.1109/IIAI-AAI50415.2020.00064.

[20] R. Rahmatulloh and R. E. Putra, "Rancang Bangun Sistem Online Judge dan Pendeteksian Plagiarisme Menggunakan Arsitektur Serverless," *Journal of Information Engineering and Educational Technology*, vol. 4, no. 1, pp. 20–28, Jun. 2020, doi: 10.26740/jieet.v4n1.p20-28.

[21] A. Ćatović, N. Buzađija, and S. Lemes, "Microservice development using RabbitMQ message broker," *Science, Engineering and Technology*, vol. 2, no. 1, pp. 30–37, Apr. 2022, doi: 10.54327/set2022/v2.i1.19.

[22] S. Hardikar, P. Ahirwar, and S. Rajan, "Containerization: Cloud Computing based Inspiration Technology for Adoption through Docker and Kubernetes," in *2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)*, IEEE, Aug. 2021, pp. 1996–2003. doi: 10.1109/ICESC51422.2021.9532917.

[23] P. Murley, Z. Ma, J. Mason, M. Bailey, and A. Kharraz, "WebSocket Adoption and the Landscape of the Real-Time Web," in *Proceedings of the Web Conference 2021*, New York, NY, USA: ACM, Apr. 2021, pp. 1192–1203. doi: 10.1145/3442381.3450063.

[24] R. Mosca, M. De Santo, and R. Gaeta, "Ontology learning from relational database: a review," *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 12, pp. 16841–16851, Dec. 2023, doi: 10.1007/s12652-023-04693-8.

[25] A. Makris, K. Tserpes, G. Spiliopoulos, D. Zissis, and D. Anagnostopoulos, "MongoDB Vs PostgreSQL: A comparative study on performance aspects," *Geoinformatica*, vol. 25, no. 2, pp. 243–268, Apr. 2021, doi: 10.1007/s10707-020-00407-w.

[26] M. Mesnier, G. R. Ganger, and E. Riedel, "Storage area networking - Object-based storage," *IEEE Communications Magazine*, vol. 41, no. 8, pp. 84–90, Aug. 2003, doi: 10.1109/MCOM.2003.1222722.

[27] N. S. Zulatifa *et al.*, "Implementasi MinIO Sebagai Sistem Penyimpanan Menggunakan Model Containerization Docker dalam Learning Management System Guru," *Journal of Computing and Informatics Research*, vol. 5, no. 1, pp. 401–409, Oct. 2025.

[28] H. Vdovic, J. Babic, and V. Podobnik, "An Application Programming Interface for Advanced Analytics of Contextually Enriched Automotive Data," in *2021 16th International Conference on Telecommunications (ConTEL)*, IEEE, Jun. 2021, pp. 70–77. doi: 10.23919/ConTEL52528.2021.9495964.

[29] M. Sholeh, I. Gisfas, Cahiman, and M. A. Fauzi, "Black Box Testing

on ukmbantul.com Page with Boundary Value Analysis and Equivalence Partitioning Methods," *J. Phys. Conf. Ser.*, vol. 1823, no. 1, p. 012029, Mar. 2021, doi: 10.1088/1742-6596/1823/1/012029.

[30] Z. M. Jiang and A. E. Hassan, "A Survey on Load Testing of Large-Scale Software Systems," *IEEE Transactions on Software Engineering*, vol. 41, no. 11, pp. 1091–1118, Nov. 2015, doi: 10.1109/TSE.2015.2445340.

[31] S. S, "A Study of Software Development Life Cycle Process Models," *SSRN Electronic Journal*, 2017, doi: 10.2139/ssrn.2988291.

[32] S. C. Hubli and Dr. R. C. Jaiswal, "Efficient Backend Development with Spring Boot: A Comprehensive Overview," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 11, no. 11, pp. 759–765, Nov. 2023, doi: 10.22214/ijraset.2023.56617.

[33] "REACT JS - A JAVASCRIPT LIBRARY," *International Research Journal of Modernization in Engineering Technology and Science*, Aug. 2024, doi: 10.56726/IRJMETS52186.