BAB VI
KESIMPULAN

A. Kesimpulan

Berdasarkan hasil dari penelitian yang telah dilakukan, maka adaptasi alat *Dolos* dan algoritma jarak *levenshtein* untuk memeriksa tingkat kemiripan proyek lunak serta pembangunan sistem informasi penunjang telah berhasil dilakukan. Algoritma yang diimplementasi telah dapat dengan baik memeriksa berbagai jenis proyek perangkat lunak seperti yang telah didefinisikan pada batasan masalah, serta sistem informasi telah dibangun dengan menggunakan prinsip *user interface* (antarmuka pengguna) dan *user experience* (pengalaman pengguna) yang modern.

Algoritma yang diimplementasi dapat menerima berbagai masukan dari pengguna terkait proses pemeriksaan yang akan dilakukan serta menerapkannya. Selain itu, proses klasterisasi berbagai proyek perangkat lunak diamati sudah dapat berjalan dengan cukup memuaskan. Terakhir, algoritma juga dapat menghasilkan laporan hasil analisis yang sesuai dengan proses pemeriksaan yang telah dilakukan.

Adapun sistem informasi yang dibangun memiliki halaman otentikasi (*login*), halaman utama, halaman tahapan algoritma (3 halaman), serta halaman laporan (4 halaman). Tampilan pada berbagai halaman ini berhasil dibuat dengan konsisten dan rapi, sehingga mempermudah pengguna dalam menggunakannya. Sistem *back-end* telah diintegrasikan dengan algoritma yang dibangun di dalam satu peladen (*server*), sehingga memudahkan komunikasi antara sistem informasi dengan algoritma.

Hasil pengujian pada algoritma menunjukkan bahwa algoritma yang diimplementasi sudah dapat dengan benar mengukur tingkat kemiripan serta mengidentifikasi klaster yang terbentuk pada jenis proyek perangkat lunak *Laravel*. Pada sisi lain, sistem informasi yang dibangun juga telah memenuhi berbagai kebutuhan fungsional menggunakan sistem pengujian *black box*. Dengan kata lain, algoritma dan sistem informasi telah dinyatakan andal pada berbagai pengujian yang dilaksanakan.

B. Saran

Berdasarkan penelitian yang telah dilakukan, masih terdapat beberapa kekurangan dan kelemahan pada algoritma dan sistem informasi. Kekurangan utama pada algoritma yang diimplementasi adalah lamanya durasi pemeriksaan disertai tingginya persentase penggunaan CPU untuk setiap pemeriksaan. Pada sistem informasi yang dibangun, walaupun memiliki tampilan yang bagus, namun terdapat beberapa konsep yang cukup sulit dijelaskan dalam bahasa Indonesia, walaupun sudah terdapat beberapa *tooltip* untuk membantu pengguna. Beberapa kekurangan dan kelemahan ini diharapkan dapat menjadi perhatian bagi peneliti selanjutnya yang ingin melakukan penelitian yang serupa.

DAFTAR PUSTAKA

[1] A. J. Mooij, J. Ketema, S. Klusener, and M. Schuts, "Reducing Code Complexity through Code Refactoring and Model-Based Rejuvenation," in *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, 2020, pp. 617–621. doi: 10.1109/SANER48275.2020.9054823.

[2] H. Rathi and R. Chatterjee, "Role and Relevance of Reuse Repository Facilitating Software Development," *SIGSOFT Softw. Eng. Notes*, vol. 39, no. 4, pp. 1–5, Aug. 2014, doi: 10.1145/2632434.2632445.

[3] B. Kaucic, D. Sraka, M. Ramsak, and M. Krasna, "Observations on Plagiarism in Programming Courses," in *International Conference on Computer Supported Education*, 2010. [Online]. Available: https://api.semanticscholar.org/CorpusID:5510198

[4] G. Cosma and M. Joy, "Source-code plagiarism : a UK academic perspective," University of Warwick. Department of Computer Science, May 2006. [Online]. Available: https://wrap.warwick.ac.uk/61520/

[5] G. Cosma *et al.*, "Perceptual Comparison of Source-Code Plagiarism within Students from UK, China, and South Cyprus Higher Education Institutions," *ACM Trans. Comput. Educ.*, vol. 17, no. 2, May 2017, doi: 10.1145/3059871.

[6] M. Joy, G. Cosma, J. Y.-K. Yau, and J. Sinclair, "Source Code Plagiarism—A Student Perspective," *IEEE Transactions on Education*, vol. 54, no. 1, pp. 125–132, 2011, doi: 10.1109/TE.2010.2046664.

[7] M. Joy and M. Luck, "Plagiarism in programming assignments," *IEEE Transactions on Education*, vol. 42, no. 2, pp. 129–133, 1999, doi: 10.1109/13.762946.

[8] H. Cheers, Y. Lin, and S. P. Smith, "Academic Source Code Plagiarism Detection by Measuring Program Behavioral Similarity," *IEEE Access*, vol. 9, pp. 50391–50412, 2021, doi: 10.1109/ACCESS.2021.3069367.

[9] R. Maertens *et al.*, “Dolos: Language-agnostic plagiarism detection in source code,” *J Comput Assist Learn*, vol. 38, no. 4, pp. 1046–1061, 2022, doi: https://doi.org/10.1111/jcal.12662.

[10] J. Shore and S. Warden, *The art of agile development*. “ O’Reilly Media, Inc.,” 2021.

[11] J. Senthilnath, S. N. Omkar, and V. Mani, “Clustering using firefly algorithm: Performance study,” *Swarm Evol Comput*, vol. 1, no. 3, pp. 164–171, 2011, doi: https://doi.org/10.1016/j.swevo.2011.06.003.

[12] I. Hooda, R. Scholar, and R. Singh Chhillar, “Software Test Process, Testing Types and Techniques,” *Int J Comput Appl*, vol. 111, no. 13, pp. 975–8887, 2015.

[13] A. E. Carpenter, L. Kamentsky, and K. W. Eliceiri, “A call for bioimaging software usability,” *Nat Methods*, vol. 9, no. 7, pp. 666–670, 2012, doi: 10.1038/nmeth.2073.

[14] R. Maertens, P. Dawyndt, and B. Mesuere, “Dolos 2.0: Towards Seamless Source Code Plagiarism Detection in Online Learning Environments,” in *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 2*, in ITiCSE 2023. New York, NY, USA: Association for Computing Machinery, 2023, p. 632. doi: 10.1145/3587103.3594166.

[15] A. Eppa and A. Murali, “Source Code Plagiarism Detection: A Machine Intelligence Approach,” in *2022 IEEE Fourth International Conference on Advances in Electronics, Computers and Communications (ICAECC)*, 2022, pp. 1–7. doi: 10.1109/ICAECC54045.2022.9716671.

[16] R. A. Rizvee, Md. Fahim Arefin, and M. B. Abid, “A Robust Objective Focused Algorithm to Detect Source Code Plagiarism,” in *2022 IEEE 13th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 2022, pp. 109–115. doi: 10.1109/UEMCON54665.2022.9965688.

[17] L. Sulistiani and O. Karnalim, “ES-Plag: Efficient and sensitive source code plagiarism detection tool for academic environment,” *Computer Applications in*

Engineering Education, vol. 27, no. 1, pp. 166–182, 2019, doi: https://doi.org/10.1002/cae.22066.

[18] J. Zhao, K. Xia, Y. Fu, and B. Cui, "An AST-based Code Plagiarism Detection Algorithm," in *2015 10th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA)*, 2015, pp. 178–182. doi: 10.1109/BWCCA.2015.52.

[19] T. D. Kesumo Siregar, "Levenshtein Distance Calculation Using Dynamic Programming for Source Code Plagiarism Checking".

[20] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: local algorithms for document fingerprinting," in *Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data*, in SIGMOD '03. New York, NY, USA: Association for Computing Machinery, 2003, pp. 76–85. doi: 10.1145/872757.872770.

[21] "jplag/JPlag: State-of-the-Art Software Plagiarism & Collusion Detection." Accessed: Jul. 02, 2024. [Online]. Available: https://github.com/jplag/JPlag

[22] L. Prechelt, G. Malpohl, and M. Philippsen, "Finding Plagiarisms among a Set of Programs with JPlag," *JUCS - Journal of Universal Computer Science*, vol. 8, no. 11, pp. 1016–1038, 2002, doi: 10.3217/jucs-008-11-1016.

[23] Kementerian Pendidikan dan Kebudayaan, "Definisi plagiat," Kamus Besar Bahasa Indonesia.

[24] G. Cosma and M. Joy, "Towards a Definition of Source-Code Plagiarism," *IEEE Transactions on Education*, vol. 51, no. 2, pp. 195–200, 2008, doi: 10.1109/TE.2007.906776.

[25] Simon, J. Sheard, M. Morgan, A. Petersen, A. Settle, and J. Sinclair, "Informing Students about Academic Integrity in Programming," in *Proceedings of the 20th Australasian Computing Education Conference*, in ACE '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 113–122. doi: 10.1145/3160489.3160502.

[26] V. T. Martins, D. Fonte, P. R. Henriques, and D. da Cruz, “Plagiarism Detection: A Tool Survey and Comparison,” in *3rd Symposium on Languages, Applications and Technologies*, M. J. V. Pereira, J. P. Leal, and A. Sim\~oes, Eds., in OpenAccess Series in Informatics (OASIcs), vol. 38. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2014, pp. 143–158. doi: 10.4230/OASIcs.SLATE.2014.143.

[27] A. Ahadi and L. Mathieson, “A Comparison of Three Popular Source Code Similarity Tools for Detecting Student Plagiarism,” in *Proceedings of the Twenty-First Australasian Computing Education Conference*, in ACE ’19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 112–117. doi: 10.1145/3286960.3286974.

[28] M. Hadjieleftheriou and D. Srivastava, *Approximate String Processing*. in Foundations and trends in databases. Now Publishers, 2011. [Online]. Available: https://books.google.co.id/books?id=_lFCJzEsfBYC

[29] “String.prototype.split() - JavaScript | MDN.” Accessed: Dec. 07, 2023. [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/split

[30] “Tree-sitter | Introduction.” Accessed: Dec. 07, 2023. [Online]. Available: https://tree-sitter.github.io/tree-sitter/

[31] “fastest-levenshtein - npm.” Accessed: Jun. 18, 2024. [Online]. Available: https://www.npmjs.com/package/fastest-levenshtein

[32] G. S. Almasi and A. Gottlieb, *Highly parallel computing*. USA: Benjamin-Cummings Publishing Co., Inc., 1989.

[33] D. Bonetta, L. Salucci, S. Marr, and W. Binder, “GEMs: shared-memory parallel programming for Node.js,” in *Proceedings of the 2016 ACM SIGPLAN International Conference on Object-Oriented Programming, Systems, Languages, and Applications*, in OOPSLA 2016. New York, NY, USA: Association for Computing Machinery, 2016, pp. 531–547. doi: 10.1145/2983990.2984039.

[34] “Cluster | Node.js v22.1.0 Documentation.” Accessed: May 08, 2024. [Online]. Available: https://nodejs.org/api/cluster.html

[35] “Child process | Node.js v22.1.0 Documentation.” Accessed: May 08, 2024. [Online]. Available: https://nodejs.org/api/child_process.html

[36] S. E. Hargitay and T. Dixon, “Categories of software,” 1991. [Online]. Available: https://api.semanticscholar.org/CorpusID:63937113

[37] E. W. Weisstein, “Cartesian Product”, Accessed: Dec. 20, 2023. [Online]. Available: https://mathworld.wolfram.com/

[38] “Information system | Definition, Examples, & Facts | Britannica.” Accessed: Dec. 20, 2023. [Online]. Available: https://www.britannica.com/topic/information-system

[39] The Editors of Encyclopaedia Britannica, “Website,” Aug. 2023. [Online]. Available: https://www.britannica.com/technology/website

[40] “Express - Node.js web application framework.” Accessed: Dec. 07, 2023. [Online]. Available: https://expressjs.com/

[41] A. Boduch, *React and React Native*. Packt Publishing, 2017. [Online]. Available: https://books.google.co.id/books?id=jLkrDwAAQBAJ

[42] G. E. Krasner, S. T. Pope, and others, “A description of the model-view-controller user interface paradigm in the smalltalk-80 system,” *Journal of object oriented programming*, vol. 1, no. 3, pp. 26–49, 1988.

[43] “The_World_of_Programming_Languages”.

[44] “HTML_XHTML_The_Complete_Reference”.