References

Aklouche, B., Bounhas, I., & Slimani, Y. (2018, November). Query Expansion Based on NLP and Word Embeddings. *27th Text REtrieval Conference (TREC)*. Retrieved April 30, 2023, from https://trec.nist.gov/pubs/trec27/papers/JARIR-CC.pdf

Asian Corporate Aviation Management Ptd. Ltd. (2022). *ACAM Flight Trip Assignment Database (February 2022 - April 2023)*. FL3XX. Retrieved March 12, 2023, from https://app.fl3xx.com/#/flights

Benefits of Aviation Management. (2023). Business Jet Consultants. Retrieved February 12, 2023, from https://businessjetconsultants.net/services/benefits-aviation-management/

@elmirap. (2017). *Official Solution - Implement Trie (Prefix Tree)*. LeetCode. Retrieved February 12, 2023, from https://leetcode.com/problems/implement-trie-prefix-tree/solutions/127843/official-solution/

Emon, R. Y., & Tista, S. C. (2019, November). An Efficient Word Lookup System by using Improved Trie Algorithm. https://doi.org/10.48550/arXiv.1911.01763

Ferré, S. (2018). Responsive and Flexible Controlled Natural Language Authoring with Zipper-Based Transformations (B. Davis, M. C. Keet, & A. Wyner, Eds.). In *Controlled Natural Language* (Vol. 304, pp. 21-30). IOS Press BV. https://doi.org/10.3233/978-1-61499-904-1-21

FL3XX. (2022). Stellar.aero. Retrieved February 12, 2023, from https://www.stellar.aero/fl3xx/

Kale, N. (2018). Improving Time and Efficiency of Trie Data Structure. MA thesis, University of Texas, Arlington. Retrieved April 13, 2023, from https://rc.library.uta.edu/uta-ir/bitstream/handle/10106/27643/KALE-THESIS-2018.pdf?sequence=1&isAllowed=y

Koppikar, U., Rajur, A., & Jayalaxmi, G. N. (2019, May 17). Efficient Word Processing Applications Using Radix Tree. *4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*, 1041-1046. https://doi.org/10.1109/RTEICT46194.2019.9016862

Kumar, M., & Bhatt, V. (2020, January). Denoising SMS Text Using Ternary Tree for FAQ Retrieval. *International Journal of Advanced Science and Technology*, *29*(3), 13200-13209. https://www.researchgate.net/publication/344191988_Denoising_SMS_Text_Using_Ternary_Tree_for_FAQ_Retrieval

Lorentson, J. (2019, June 18). *Autocomplete suggestions: varieties, benefits & UX best practices*. Fresh Consulting. Retrieved February 12, 2023, from https://www.freshconsulting.com/insights/blog/autocomplete-benefits-ux-best-practices/

Malviya, A., Verma, P., Purohit, D. K., & Singh, V. (2018, January). Predictive auto-completion for query in search engine. *International Journal of Business Information Systems (IJBIS)*, *28*(3), 299-314. https://doi.org/10.1504/IJBIS.2018.10013682

Mishra, H. M. B. (2023). *Autocomplete feature using TRIE Data Structure*. OpenGenus IQ. Retrieved February 12, 2023, from https://iq.opengenus.org/autocomplete-using-trie-data-structure/

Price, C.E. (1971). Table Lookup Techniques. *ACM Comput. Surv.,* 3, 34-64. https://dl.acm.org/doi/pdf/10.1145/356586.356587

Putri, M. D. A. (2022). Implementation of Compact Trie in ReactAutocomplete Input Component to Increase MemoryEfficiency. https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2022-2023/MakalahMatdis2022.htm

Roseman, M. (2020, February 3). *Autocomplete Radix Tree (Trie) in Javascript*. Matt Roseman's Blog. Retrieved February 7, 2023, from https://mroseman.com/blog/autocomplete-radix-tree

Vivien, L. (2022, July 18). *Autocomplete with trie (3 solutions) - Java JavaScript Python*. La Vivien Post. Retrieved March 12, 2023, from https://www.lavivienpost.com/autocomplete-with-trie-code/

Wangmo, C., & Wiese, L. (2022, July 11). Efficient Subgraph Indexing for Biochemical Graphs. *Proceedings of the 11th International Conference on Data Science, Technology and Applications - DATA*, 533-540. https://doi.org/10.5220/0011350100003269

Wu, G. (2023, March 11). *Hash Table Vs. Trie (Prefix Tree)*. Baeldung. Retrieved March 12, 2023, from https://www.baeldung.com/cs/hash-table-vs-trie-prefix-tree

Ye, Y. (2019). *Data Structures*. INDIANA UNIVERSITY. Retrieved March 12, 2023, from https://cgi.luddy.indiana.edu/~yye/c343-2019/tries.php

Yulianto, M. M., Arifudin, R., & Alamsyah, A. (2018, May). Autocomplete and Spell Checking Levenshtein Distance Algorithm To Getting Text Suggest Error Data Searching In Library. *Scientific Journal of Informatics*, *5*(1), 67-75. https://doi.org/10.15294/sji.v5i1.14148