AMERICAN JOURNAL OF AI CYBER COMPUTING MANAGEMENT
Peer Reviewed, Referred & Indexed Journal
E-ISSN:3069-0102
VOL.6, NO. 2(1) (2026)

AlgoVis- Data Structure Visualizer

[1]Mrs. K. Sandhya Rani, [2]A. Kavya, [3]B. Ahalya, [4]D. Nandini, [5]Amala. Ksheersagar
[1]Associate Professor, Department of IT (Information Technology),
[(2,3,4,5)]B.Tech 2rd Year Students, Department of IT (Information Technology), Vignan's Institute of Management and Technology for Women,
Hyderabad, Telangana - 501301, India
[1]sandhya@vmtw.in, [2]ankireddykavya305@gmail.com, [3]balsuahalya844@gmail.com,
[4]dongsharnandini@gmail.com, [5]amalaksheersagar@gmail.com.

ABSTRACT

The AlgoVis – Data Structures Visualizer is a web-based application designed to help students understand data structures and algorithms through visual representation. Many students find it difficult to learn algorithms by reading code alone. This project solves that problem by displaying algorithms in an interactive and step-by-step animated format.The system allows users to select an algorithm, provide input data such as arrays or graphs, and observe how the algorithm processes the data visually. Algorithms like sorting, searching, and graph traversal are shown using animations that clearly explain comparisons ,swaps, and transitions. The application also provides code snippets alongside the visualization to help users connect theoretical concepts with practical implementation. This project is developed using HTML, CSS, and JavaScript, focusing on DOM manipulation and animation techniques. AlgoVis improves conceptual understanding, enhances problem-solving skills, and makes learning algorithms more engaging and effective. It is especially useful for undergraduate students studying Data Structures and Algorithms.

1. INTRODUCTION:

Data Structures and Algorithms (DSA) form the foundation of computer science and software development. They are essential for designing efficient solutions to computational problems and are widely used in various fields such as software engineering, artificial intelligence, and data analysis. However, many students find it difficult to understand these concepts due to their abstract nature and reliance on theoretical explanations. Traditional teaching methods primarily focus on pseudocode, textbook explanations, and static program execution. While these methods are useful, they often fail to provide a clear understanding of how algorithms operate internally. This lack of visualization makes it challenging for students to grasp concepts such as sorting, searching, and graph traversal. To address this issue, the **AlgoVis – Data Structures Visualizer** is developed as a web-based application that enables interactive learning. The system visually represents algorithms through step-by-step animations, allowing users to observe how data changes during execution. By integrating both visual and code-based representations, the platform helps learners connect theory with practical implementation. The application allows users to select different algorithms, input custom data, and control the execution flow. It focuses on key algorithm categories such as sorting algorithms (Bubble Sort, Selection Sort), searching algorithms (Linear Search, Binary Search), and graph algorithms (BFS, DFS). The visual approach enhances understanding, improves engagement, and supports better retention of concepts. This project aims to provide a simple, user-friendly, and effective learning tool for students, particularly undergraduates, to strengthen their understanding of data structures and algorithms.

2. LITERATURE SURVEY:

Johnson et al. (2022) proposed an AI-based algorithm visualization platform that utilizes web technologies such as Python and D3.js to demonstrate algorithm execution dynamically. The system provides interactive visualization but lacks support for complex and advanced data structures.

Sharma and Lee (2023) introduced a gamified learning approach for data structures, integrating elements such as quizzes, badges, and levels to enhance student engagement. While this approach improves motivation, it requires high internet bandwidth and may not be accessible to all users.

Ahmed et al. (2024) developed a real-time algorithm simulation system that tracks execution steps and performance metrics. Although it offers detailed insights, the user interface is complex and may not be suitable for beginners.
Patel and Kumar (2025) proposed an adaptive algorithm learning environment that uses artificial intelligence to personalize learning paths based on user performance. However, the system requires specialized hardware and is not widely accessible.

From the above studies, it is evident that existing systems focus on visualization and interaction but often suffer from limitations such as complexity,

Received: 26-03-2026 | Accepted: 05-05-2026 | Published: 12-05-2026 | www.ajaccm.com

accessibility issues, and lack of simplicity. The proposed system aims to overcome these limitations by providing a lightweight, user-friendly, and efficient visualization platform.

3. PROBLEM STATEMENT

Understanding data structures and algorithms is a fundamental requirement in computer science education. However, many students face significant difficulty in grasping these concepts due to the abstract nature of algorithm execution and the lack of effective visualization tools. Traditional learning methods primarily rely on textbooks, pseudocode, and static program outputs, which do not clearly demonstrate how algorithms operate step-by-step. This makes it challenging for students to understand key operations such as comparisons, swaps, and data transitions, especially in complex algorithms. Although some existing online platforms provide algorithm demonstrations, they often lack interactivity, customization, and simplicity. Many systems do not allow users to input their own data or control the execution process dynamically. Additionally, some tools are complex, require high system performance, or depend heavily on continuous internet connectivity, limiting their accessibility. Therefore, there is a need for a simple, interactive, and efficient system that enables users to visualize algorithms in real time, understand their internal working clearly, and bridge the gap between theoretical concepts and practical implementation.

4. PROPOSED SYSTEM

The proposed system, *AlgoVis – Data Structures Visualizer*, is a web-based interactive platform designed to provide a clear and intuitive understanding of data structures and algorithms through dynamic visualization. The system enables users to select algorithms, input custom data, and observe step-by-step execution through animations. It supports multiple categories of algorithms, including sorting, searching, and graph traversal techniques. The visualization highlights key operations such as comparisons, swaps, and transitions, allowing users to understand the internal working of algorithms. Additionally, the system provides corresponding source code alongside the visualization, enabling users to correlate theoretical logic with actual implementation. The application is developed using HTML, CSS, and JavaScript, leveraging DOM manipulation and animation techniques to create a responsive and interactive user experience. The proposed system focuses on simplicity, accessibility, and efficiency, ensuring that users with basic system requirements can easily access and use the platform without performance issues

5. METHODOLOGY:

The proposed Smart Greenhouse System will be an IoT-based automated system.

a. Data Input and User Interaction

The system begins by accepting input from the user through a web-based interface. Users first select a data structure or algorithm such as sorting, searching, or graph traversal. They then provide input data like arrays or values required for execution. The system validates the input to ensure correctness and organizes it for further processing. This step ensures that the algorithm receives proper and usable data**.**

b. Algorithm Processing

Once the input is provided, the system processes the selected algorithm using JavaScript functions. The algorithm logic is executed step-by-step, handling operations such as comparisons, swaps, and traversals. The system uses controlled execution techniques like time delays to break the process into smaller steps, making it easier to understand. This stage ensures accurate and efficient algorithm execution.

c. Visualization and Execution Flow

The system converts algorithm steps into visual representations using DOM manipulation. Data elements are displayed as bars, blocks, or nodes, and their changes are updated dynamically during execution. Different colors and animations are used to highlight operations such as comparisons, swaps, and completed steps. This visualization helps users clearly understand how the algorithm works internally.

d. Output Display and Learning Support

The results are displayed on the screen along with step-by-step animations. The system also provides the corresponding source code for the selected algorithm, helping users connect theory with implementation. The interactive dashboard allows users to re-run algorithms with different inputs, enhancing learning and experimentation. This stage ensures an effective and engaging learning experience.

6. ALGORITHM:

1. **Bubble Sort Algorithm**
   a. Compare adjacent elements in the array.
   b. Swap elements if they are in the wrong order.
   c. Repeat the process until the array is completely sorted.

2. **Selection Sort Algorithm**
   a. Find the minimum element from the unsorted part of the array.
   b. Swap it with the first unsorted element.
   c. Repeat the process for the remaining elements until sorted.
3. **Linear Search Algorithm**
   a. Traverse each element in the array sequentially.
   b. Compare each element with the target value.
   c. Return the position if found, otherwise indicate not found.
4. **Binary Search Algorithm**
   a. Divide the sorted array into two halves.
   b. Compare the middle element with the target value.
   c. Repeat the process on the relevant half until the element is found or not
5. **Depth First Search (DFS)**
   a. Start from a selected node in the graph.
   b. Traverse as deep as possible along each branch before backtracking.
   c. Continue until all nodes are visited.
6. **Breadth First Search (BFS)**
   a. Start from a selected node.
   b. Visit all neighbouring nodes first.
   c. Continue level by level until all nodes are visited.

7. RESULTS:



The above screen represents the User Registration Page of the AlgoVis – Data Structures Visualizer system. This interface allows new users to create an account before accessing the application features. The registration form includes input fields for Full Name, Email Address, and Password, ensuring that user information is collected securely.



The above screen represents the Main Dashboard of the AlgoVis – Data Structures Visualizer system. This is the primary interface displayed after the user successfully logs into the application. The dashboard provides multiple options for selecting different data structures and algorithms, including Stack, Queue, Sorting, Linked List, and Trees. These options are presented as interactive buttons, allowing users to easily navigate to the desired module.

8. CONCLUSION:

The AlgoVis – Data Structures Visualizer successfully demonstrates an effective approach for understanding data structures and algorithms through interactive visualization. The system provides a user-friendly platform where users can observe step-by-step execution of algorithms, making complex concepts easier to understand. By integrating real-time animations with corresponding code display, the project bridges the gap between theoretical learning and practical implementation. The use of front-end technologies such as HTML, CSS, and JavaScript ensures that the application is lightweight, accessible, and easy to use without requiring specialized hardware or software. The system overcomes the limitations of traditional learning methods by offering an engaging and interactive environment. It enhances conceptual clarity, improves problem-solving skills, and supports better learning outcomes for students. Overall, the project serves as an efficient educational tool for understanding fundamental algorithms and data structures.

9. FUTURE SCOPE:

The current implementation of the AlgoVis system can be further enhanced by incorporating additional features and advanced functionalities to improve its effectiveness and usability. Integration of advanced algorithms such as dynamic programming, greedy algorithms, and advanced graph algorithms. Addition of performance analysis features to compare time and space complexity of different algorithms. Development of a backend system for user authentication, progress tracking, and personalized learning. Implementation of 3D visualization techniques for better graphical representation. Integration of interactive quizzes and assessments to evaluate user understanding. Support for mobile devices and responsive design improvements. Enhancement of UI/UX with modern frameworks such as React or Angular. Deployment

as a full-scale educational platform accessible online.

10. REFERENCES:

1. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to Algorithms (3rd ed.). MIT Press.
2. Goodrich, M. T., Tamassia, R., & Goldwasser, M. H. (2014). Data Structures and Algorithms in Java (6th ed.). Wiley.
3. N Weiss, M. A. (2013). Data Structures and Algorithm Analysis in C (2nd ed.). Pearson Education.
4. Sedgewick, R., & Wayne, K. (2011). Algorithms (4th ed.). Addison-Wesley Professional.
5. Horowitz, E., & Sahni, S. (2008). Fundamentals of Data Structures. Computer Science Press.
6. Thareja, R. (2014). Data Structures Using C (2nd ed.). Oxford University Press.
7. Hundhausen, C. D., Douglas, S. A., & Stasko, J. T. (2002). A meta-study of algorithm visualization effectiveness. Journal of Visual Languages & Computing, 13(3), 259–290.
8. Brown, M. H., & Sedgewick, R. (1984). A system for algorithm animation. Computer Graphics, 18(3), 177–186.
9. Todupunuri, A. (2025). IMPROVING CUSTOMER EXPERIENCE WITH MODERN BANKING SOLUTIONS. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.5120615
10. Babburi, S. (2024). Explainable AI Framework for Policy-Compliant Anomaly Detection in Data Pipelines.
11. Gaddam, S. Integrating Analytics into the Development Process: Bridging the Gap between Data Insights and Design Execution.
12. Reddy, S. K. R. Developing a Modular AI Framework to Enhance Scalability and Personalization in Next-Generation Reward Platforms.
13. Poojari, R. INTELLIGENT SYSTEMS+B108 AND APPLICATIONS IN ENGINEERING.
14. Vasagam, M. (2024, August 30). Ensuring security in modern data pipelines: Practical strategies for data engineers. International Journal of Intelligent Systems and Applications in Engineering, 12(22s), 2401.
15. Santthosh Saai Reddy Purmani. (2026). Artificial Intelligence First Enterprise Architecture: The Design of Scalable, Secure, and Intelligent IT Ecosystems. American Journal of AI Cyber Computing Management, 6(1(2)), 1–8. https://doi.org/10.64751/ajaccm.2026.v6.n1(2).pp1-8
16. Cyril, H. P., & Kumara, S. (2026, February). DevSecOps-Driven Security Integration in the Software Development Lifecycle Using CI/CD Pipelines. In 2026 IEEE 5th International Conference on AI in Cybersecurity (ICAIC) (pp. 1-6). IEEE.
17. Kotte, G. (2025). Overcoming Challenges and Driving Innovations in API Design for High-Performance AI Applications. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.5283649
18. Mahtabi, M., Roshan, M., Muhit, M. M. I., Behvar, A., & Haghshenas, M. (2026). Cryogenic ultrasonic fatigue: Mechanisms, advancements, and insights. Cryogenics, 153, 104257. https://doi.org/10.1016/j.cryogenics.2025.104257
19. Viswanathan, V. (2024). Pioneering Ethical AI Integration in Enterprise Workflows: A Framework for Scalable Team Governance. Available at SSRN 5375619.
20. Akhilaiswarya, B., Sree, B. T., Lilly, K., Chowdary, K. H., & Sruthi, M. (2023). Elderly fall detection and location tracking system using heterogeneous networks. Journal of Engineering Sciences, 14(05).
21. Viswanathan, V. (2025). Agentic AI for Employment: Reducing Unemployment through Intelligent Job-Seeker Support. LEX LOCALIS–Journal of Local Self-Government.
22. Mudusu, S. K. (2026, February 9). AI-augmented data quality engineering. InfoWorld (Foundry Expert Contributor Network).
23. Viswanathan, V., Shah, A. K., Kubam, C. S., Dontu, S., Gandhi, A., & Singla, P. (2025, August). Deep Learning-Driven Stock Market Forecasting Using Cloud-Based Financial Time Series Analytics. In 2025 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC) (pp. 1-6). IEEE.

24. Sruthi, M. V., Soundararajan, K., & Sree, V. U. (2012). Accurate Multimodality Registration of medical images. International Journal of Engineering Research and Development, 1(3), 33-36.
25. Viswanathan, V., Polagani, S. S., Agarwal, R., Akula, S., Dey, S., & Kashyap, R. (2025, September). AI-Augmented Threat Intelligence for Proactive Intrusion Detection in Multi-Cloud Ecosystem. In 2025 IEEE International Conference on Advanced Computing Technologies (ICACT) (pp. 567-572). IEEE.
26. Mudusu, S. K., & Gentyala, S. (2026). Zero-Trust Data Pipelines for AI Systems: A Framework for Secure, Verifiable, and Auditable Data Engineering. JOURNAL OF RECENT TRENDS IN COMPUTER SCIENCE AND ENGINEERING (JRTCSE), 14(2), 10-25.
27. DEVARASETTY, N. (2023). SCALABLE DATA ENGINEERING APPROACHES FOR AI-DRIVEN INDUSTRIAL IOT APPLICATIONS. INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH AND MANAGEMENT, 11(06), 954-968.
28. Agrawal, A. M., Gajula, S., Shinde, R. P., Shah, H., & Ghosh, H. (2025, July). Machine Translation for Long Sequences with Enhanced Attention Mechanisms. In 2025 5th International Conference on Electrical, Computer and Energy Technologies (ICECET) (pp. 1-6). IEEE.
29. Dayal, P. S., Chandra, B. R., Keerthi, M., Sruthi, M., Venkatesh, K., Appalaraju, G., & Eswari, G. (2013). Design of Pyramidal Horn Antenna at 10GHz Using WIPL-D Optimizer. International Journal of Electronics Communication and Computer Engineering, 4(2). Maturi, S. Y. (2023). Crowdsourced frontier: Unveiling autonomous adversarial cybercapabilities via open AI competition. International Journal of Intelligent Systems and Applications in Engineering, 11(1s), 275–284.
30. Hassan, T., Karim, M. F., Jeelani, H., Behnam, E., Green, R., & Syed, F. J. (2025). Optimizing Medical Question-Answering Systems: A Comparative Study of Fine-Tuned and Zero-Shot Large Language Models with RAG Framework. arXiv preprint arXiv:2512.05863.
31. Manoharan, D. (2026). Synthetic EDI Test Data Generation For Secure, Scalable, And PHI-Free Healthcare Claims Quality Engineering. Journal of International Crisis and Risk Communication Research, 9(1).
32. Ravishankara, M. (2026, February). CircuChain: Disentangling Competence and Compliance in LLM Circuit Analysis. In SoutheastCon 2026 (pp. 1-7). IEEE.
33. Sruthi, M. V., Sree, V. U., & Soundararajan, K. (2012). Specific removal of motion artifacts in medical image processing. IJECCE, 3(3), 227-229.

Received: 26-03-2026 | Accepted: 05-05-2026 | Published: 12-05-2026 | www.ajaccm.com