Algorithm Visualization Project Report

Instructor: Prof. Shereen Taie

TA: Eng. Ismail Ahmed

Team Members

Karim Bassem Joseph (ID: 231000797) – Project Lead & Backend Developer

Nadeem Diaa Shokry (ID: 231000857) – Frontend Developer

Ahmed Khaled Shalaby (ID: 231000514) – Algorithm Specialist and UI/UX

Mahmoud Ahmed Farag (ID: 231002115) – Algorithm Specialist and Testing

Introduction

Algorithms are an essential part of computer science and software development. However, understanding how algorithms work internally can sometimes be difficult, especially for beginners.
Our project, Algorithm Visualizer, aims to solve this problem by providing an interactive web application where users can see how sorting algorithms behave step-by-step. By visualizing the process, users can better understand the logic behind each algorithm.

Project Objectives

The main goals of our project are:

1. Build a simple and intuitive web interface for visualizing sorting algorithms.
2. Develop animations that show each step of the algorithm.
3. Allow users to input their own datasets and watch how the algorithm handles them.
4. Provide performance comparisons (time and space complexity) between different algorithms.

Technologies Used

- Backend: Python (Flask Framework)
- Frontend: HTML, CSS, JavaScript
- Algorithms: Implemented in Python (Bubble Sort, Insertion Sort, Selection Sort, Merge Sort, Quick Sort)

Project Features

1. How the System Works

The frontend sends the user's dataset to the Flask backend. The backend then runs the selected algorithm and records each step in detail—every comparison, swap, and change—in a structured dictionary. This data is sent back to the frontend as a JSON file. JavaScript on the frontend reads the JSON and creates animations that show exactly what happened during sorting.

2. Algorithm Visualization

We built visualizations for five common sorting algorithms. The user can watch how the algorithm handles the dataset, step by step, with clear visuals showing comparisons and swaps.

3. Interactive User Controls

Users can:

- Start, pause, or step through the animation.
- Choose the speed of the visualization.
- Enter their own data to see how it is sorted.

4. Educational Tools

To support learning, we added:

- Descriptions of each algorithm's time and space complexity.
- A feature to compare different algorithms based on performance.

5. User Experience

The web app is:

- Clean and responsive (it works well on phones, tablets, and computers).
- Easy to use, with smooth navigation.
- Built with proper error handling to manage invalid inputs.

Conclusion

Through this project, we combined our knowledge of algorithms, backend development, and frontend design to create an educational tool that helps users visualize sorting processes easily.
We successfully achieved the objectives we planned, and the system is working as intended.
This project also helped us improve our teamwork, problem-solving, and software development skills.

We believe Algorithm Visualizer can be a useful tool for students who want to build a better understanding of sorting algorithms.