A

PROJECT REPORT ON

"ALGORITHM VISUALIZER"

Submitted in partial fulfilment of requirements for the award of degree of

Bachelor of Technology

In

Computer Engineering

by

1. Niraj J. Tiwari (2010121245042)

2. Snehal V. Ingole (1910121245069)

3. Namita S. Akarte (1910121245042)

4. Ganesh P. Ingale (1910121245064)

Under the Guidance of

Prof. Sudesh A. Bachwani

Department Of Computer Engineering

Government College Of Engineering, Yavatmal

2022-2023

Department Of Computer Engineering

Government College Of Engineering, Yavatmal

(An Institute of Government of Maharashtra)

CERTIFICATE

This is to certify that the Project report entitled

"**ALGORITHM VISUALIZER**"

is a bonafide project work and has been carried out by team:

Niraj j. Tiwari	(2010121245042)
Snehal V. Ingole	(1910121245069)
Namita S. Akarte	(1910121245042)
Ganesh P. Ingale	(1910121245064)

of Final Year B-Tech class under the guidance of Prof. Sudeesh A. Bachwani during the academic year 2022 -2023(Sem-VIII).

Prof. S. A. Bachwani
Project Guide

Prof. C. V. Andhare
Head, Computer Department

Dr. P. M. Khodke
Principal

Department Of Computer Engineering

Government College Of Engineering, Yavatmal

(An Institute of Government of Maharashtra)

This is to certify that the Project report entitled

"**ALGORITHM VISUALIZER**"

is a bonafide project work submitted by:

Niraj J. Tiwari	(2010121245042)
Snehal V. Ingole	(1910121245069)
Namita S. Akarte	(1910121245042)
Ganesh P. Ingale	(1910121245064)

in partial fulfillment for the award of degree of bachelor of technology In Computer Engineering.

Prof. S. A. Bachwani
Project Guide

Prof. C. V. Andhare
Head, Computer Department

Internal Examiner

External Examiner

ABSTRACT

Algorithm analysis and design is a great challenge for both computer and information science students. Fear of programming, lack of interest and the abstract nature of programming concepts are main causes of the high dropout and failure rates in introductory programming courses. It has been observed that learning using visual aids is less complicated than learning using traditional methods keeping this fact in mind we have come up with 'Algorithm Visualizer'. The aim of the project is to build a website that enables a user to visualize the steps and iterations involved in Data Structures and Algorithms using custom input and various animations. Algorithm Visualizer is an easy-to-set-up and fully automatic visualization system with step-by-step explanations of algorithms.

Through our work we intend to change the perception of this subject from complex and hard to grasp, to interesting and fun. Our work aims to engage the students by providing self-paced hands-on experience, fun filled games through mazes and patterns and interactive, perceivable visualizations for their better concept understanding of various algorithms. Our work presently focuses on path-finding, sorting and CPU scheduling algorithms as these are the most widely taught and used algorithms in the computer science domain. Our idea would not only help the students and learners get a better hold on the concepts of algorithms but also provide an innovative way for teachers and educators to portray their ideas more clearly and interactively through to the students.

Keywords— Algorithm Visualization, Pathfinding, Sorting, Recursive sorting, Recursion tree, Searching, Binary search, Turing machine, Prime number, Scheduling, Stack, Queue, Tree, N-queen

ACKNOWLEDGEMENT

Apart from individual efforts, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental throughout the project work.

It is our privilege to express our gratitude towards our project guide, **Prof. S. A. Bachwani**, for their valuable guidance, encouragement, inspiration and whole-hearted cooperation throughout the project work. We thank him for being a motivation through all our highs and importantly, our lows.

We deeply express our sincere thanks to our Head of Department **Prof. C. V. Andhare** for encouraging and allowing us to present the project on the topic "Intelligent Grain Storage Management System" and providing us with the necessary facilities to enable us to fulfill our project requirements as best as possible. We take this opportunity to thank all faculty members and staff of Department of Information Technology, who have directly or indirectly helped our project.

We pay our respects to honorable Principal **Dr. P. M. Khodke** for their encouragement. Our thanks and appreciations also go to our family and friends, who have been a source of encouragement and inspiration throughout the duration of the project.

Table of Contents

List of Figure

Chapter 1 Introduction

1.1 Introduction

Algorithm Visualizer (also referred as algorithmic animation) uses dynamic graphics to see computation of a given algorithmic program. First makes an attempt to robustness of algorithms date to mid80's (Brown, 1988; Brown and Sedgewick, 1985), and also the golden age of algorithmic visualization was around the year 2000, when outstanding software tools for a dynamic algorithmic visualization (e.g., the language Java and its graphic libraries) and sufficiently powerful hardware were already available on the market. It had been expected that algorithmic visualization would replace the way algorithms are taught. Many algorithmic animations had appeared, largely for straight forward problems like basic tree data structures and sorting. There have been even attempts to automatize development of animated algorithmic program and algorithmic visualization. Another direction was to develop tools that might permit students to make their own animations simply. Rather than giving explicit references to algorithmic animation papers, the reader is directed to a super-reference (Algoviz) that brings a listing over 700 authors, a number of them even with twenty-nine references in algorithmic animation and visualization. There are also various web pages that supply algorithmic animation systems, e.g. Algoanim, Algomation, DD2, AlgoLiang, VisuAlgo.

When we talk about complex subject topics like Algorithms, it becomes extremely necessary for students to have a strong grip over the topic as it would form the foundation of their computational thinking and programming skills. We had observed that through conventional methods of teaching it becomes a little difficult for students to understand the concept and also for teachers to explain their thoughts.

Motivated by the age-old saying, "a picture speaks more than thousand words", many researchers and educators assume that students would learn an algorithm faster and more thoroughly using algorithm visualization techniques1. So, we developed a method of learning through visualization and hand-on experience over different searching and sorting algorithms which is bound to help the students and teachers. Good visualizations bring algorithms to life by graphically representing their various states and animating the transitions between those states, especially dynamic algorithm visualization which shows a continuous, movielike presentation of an algorithm's operations. Visualization allows the human visual system to extend human intellect; we can use it to better understand these important conceptual processes, other things, too.

Also, we are well aware of the fact that the more we do things ourselves and engage the more we tend to learn about a particular topic. Thus, engaging in various game like activities can surely help the users get a hold on the topics.

The objective of this study is to style a system of algorithmic visualization, implement the system and visualize the run time for every implemented algorithmic programme aims to assist students acknowledge algorithms additional effectively.

1.2 Motivation

The motivation behind Algorithm Visualizer is to improve the teaching and learning of algorithms by providing an interactive and engaging visualization platform. Algorithms are an essential part of computer science, and understanding them is crucial for solving real-world problems. However, algorithms can be complex, abstract, and difficult to understand, particularly for novice learners.

Algorithm Visualizer was developed to address this challenge by providing a visual representation of algorithm execution, making it easier for learners to understand their logic and functionality. The motivation behind Algorithm Visualizer is to make algorithm learning accessible and engaging for learners of all levels, from novice learners to advanced researchers.

Algorithm Visualizer is motivated by the following factors:

Lack of visual aids: Traditional teaching methods for algorithms often rely on textual explanations and code examples, which can be difficult to follow and understand. Algorithm Visualizer addresses this issue by providing a visual representation of algorithm execution, making it easier for learners to understand their behavior and functionality.

Need for interactivity: Algorithms can be challenging to understand without hands-on experience. Algorithm Visualizer addresses this issue by providing an interactive platform that allows learners to experiment with different inputs, adjust complexity levels, and visualize algorithm execution step by step.

Importance of collaboration: Learning algorithms is a collaborative process, and learners often benefit from working together and sharing their experiences. Algorithm Visualizer provides a platform for sharing algorithms and their visualizations, enabling learners to collaborate and learn from each other.

Desire for innovation: Algorithm Visualizer is motivated by a desire to innovate and improve the teaching and learning of algorithms. By providing a user-friendly and interactive platform, Algorithm Visualizer aims to enhance learners' understanding and engagement with algorithms and support research in algorithm visualization.

Overall, the motivation behind Algorithm Visualizer is to make algorithm learning accessible, engaging, and collaborative for learners of all levels.

1.3 Scope

It has been found that it becomes easier for humans to retain the concepts when learnt through visuals than just textual or speech explanations.

Application is extremely user friendly so people of any age can engage and start learning new things right away. The application would also include various fun filled activities like visualization through mazes and patterns.

This application will also include a parameter of time complexity which will be displayed after the particular sorting algorithm has completed its execution for better comparison.

Almost all the famous and important algorithms will be present in the application for visualization with both path-finding and sorting algorithms present in same application, thus making it a one stop destination for the students of this domain

With the inclusion of the proposed features, the algorithm would not only help in better visualization and retention of concepts by students, but also, enable the students have a gradual and smooth transition from school level to college level of education, thus, enhancing their knowledge and productivity.

1.4 Basic concept

The basic concept of Algorithm Visualizer is to provide a web-based platform for visualizing the execution of algorithms. Algorithms are a set of instructions that are executed to solve a specific problem. Algorithm Visualizer makes it easier for users to understand how algorithms work by displaying their execution in a visual form.

Algorithm Visualizer works by breaking down the algorithm's execution into a series of steps, and then visualizing each step using various visual elements such as graphs, charts,

and animations. Users can interact with the visualization, pause, rewind, and fast-forward the execution, and explore the data structures used in the algorithm.

Algorithm Visualizer provides a user-friendly and interactive interface, making it easier for users to experiment with different algorithms, input data, and complexity levels. It also includes an algorithm library, which provides a collection of algorithms with their descriptions, complexity levels, and implementations, making it easier for users to explore and learn about different algorithms.

Algorithm Visualizer is designed to improve users' understanding and learning of algorithms, increase engagement and collaboration, and support research in algorithm visualization. It is useful for students, teachers, researchers, and developers who want to understand and learn algorithms visually.

1.5 Objective

Improve understanding: The primary objective of Algorithm Visualizer is to improve users' understanding of algorithms by providing a visual representation of their execution. Users can see how the algorithm works step by step, making it easier for them to understand the algorithm's logic and functionality.

Enhance learning: Algorithm Visualizer aims to enhance users' learning by providing a platform for experimentation and exploration. Users can input different data sets and adjust the complexity levels to see how the algorithm behaves in different scenarios.

Increase engagement: By providing a user-friendly and interactive interface, Algorithm Visualizer aims to increase users' engagement in learning algorithms. Users can interact with the visualization, pause, rewind, and fast-forward the execution, and explore the data structures used in the algorithm.

Promote collaboration: Algorithm Visualizer provides a platform for sharing algorithms and their visualizations, enabling users to collaborate and learn from each other. The algorithm library provides a collection of algorithms with their descriptions, complexity levels, and implementations, making it easier for users to explore and learn about different algorithms.

Support research: Algorithm Visualizer aims to support research in algorithm visualization by providing a platform for testing and evaluating new visualization techniques and algorithms. Researchers can use Algorithm Visualizer to conduct user studies and evaluate the effectiveness of different visualization techniques.

Overall, the objectives of Algorithm Visualizer are to enhance users' understanding and learning of algorithms, increase engagement and collaboration, and support research in algorithm visualization.

When we learn various algorithms whether it be from data structures or any basic geometry, most of the tools explain them without providing any details about how they are actually represented and their behavior in a real program. This results in the students knowing only how the data structures work theoretically and may not be able to use them practically for solving a programming or any other task which in turn further enlarges the gap between students' theoretical and practical ability, adversely affecting the students' problem-solving skill.

1.6 Software/ Hardware Requirement

Software Requirement:

- **Windows 7 or above**
- **Npm version 6.14.18**
- **Visual Studio Code**
- **Node.js version 18.16.0**

Hardware Requirement:

- **Laptop or desktop**
- **4GB RAM or Higher**

Chapter 2 Literature Review

2.1 Overview

Algorithm visualizer is an online platform that allows users to visualize how different algorithms work. It is a useful tool for students and professionals who want to understand how algorithms work, and how to improve them. The visualizer can help users learn how to solve complex problems and optimize algorithms.

The platform provides a visual representation of how an algorithm works, step by step. Users can input their own data and see how the algorithm processes it. They can also adjust various parameters of the algorithm to see how it affects the outcome.

Algorithm visualizer supports a variety of algorithms, including sorting, searching, graph traversal, dynamic programming, and more. Each algorithm has a unique visualization that helps users understand how it works.

Algorithm visualizer is a great tool for anyone who wants to learn more about algorithms and how they work. It provides an interactive and engaging way to learn, making it easier to understand complex concepts.

2.2 Literature Survey

In addition to many hundreds of individual Algorithm Visualizers created over the years, there have also been many systems created to support Algorithm Visualization development. A significant amount of research has been devoted to understanding the pedagogical effectiveness of Algorithm Visualizers. We examine some of the literature in this section.

Relevance Of Work

Every software engineer should have a good understanding of DSA to develop efficient software. Visualizers have a good history of providing effective understanding to the users. Many algorithm visualizers have been developed over the years.

In 2008, paper “AlCoLab: Architecture of Algorithm Visualization System” concerns the style of script supported algorithm visualization systems for educational purposes, focusing on the support and the improvement that those systems provide in the process of teaching of an conceptual subject such as algorithms.

In 2019, paper “Towards Developing an Effective Algorithm Visualization Tool for Online Learning” reports a work-in-progress research project at Athabasca University on

developing an effective algorithm visualization tool for online learning.

In 2019, paper “Open Interactive Algorithm Visualization” presents a work-in-progress project form developing an open interactive algorithm visualization website.
In 2021, paper “AlgoAssist: Algorithm Visualizer and Coding Platform for Remote Classroom Learning” focuses on "algorithm visualization", which allows a better understanding of its flow and operation. It supports the combination of the lab into a single application dedicated to pre-assessment, algorithm explanation, visualization, coding, and post-assessment.

In 2021, paper “Algorithm Visualizer” aims to simplify and deepen the understanding of algorithms operation. Within the paper we talk about the possibility of improving the standard methods of teaching algorithms, with the algorithm visualizations.

Pedagogical Effectiveness of Algorithm Visualizations

Algorithm Visualizers can provide a compelling alternative to other types of instruction, particularly written presentations such as pseudocode and real code. In lecture situations, students routinely report liking Algorithm Visualizers [Gurka and Citrin 1996; Stasko et al. 2001], and faculty appear to strongly support them in principle [Naps et al. 2002]. The literature, however, has not demonstrated that Algorithm Visualizers are always effective in practice. Results formed a continuum from “no significant difference” [Gurka and Citrin 1996; Hundhausen and Douglas 2000; Jarc et al. 2000] to demonstrations that Algorithm Visualizers can indeed improve understanding of data structures and algorithms [Shaffer et al. 2007; Lawrence et al. 1994; Hansen et al. 2000; Hundhausen et al. 2002]. The existing body of research helps to illuminate some of the features of Algorithm Visualizer that make them effective.

Gurka and Citrin [1996] found little to no evidence that visualizations are effective teaching tools in a survey of previous experiments. They point out that while many fields are concerned with false positives in evaluation, the visualization community may need to concern itself with false negatives. They suggest repeating some “failed” experiments from previous projects with tighter controls on issues such as usability, algorithm difficulty, animation quality, and experimental design. They caution, however, that the community must be prepared to accept an ultimate finding of “no significant difference.” Hundhausen and Douglas [2000] found that students who construct their own visualizations might be distracted by the creation process. Their experiment was based on the finding that learner involvement increases effectiveness, but technology does not necessarily improve the result.

Chapter 3 Problem Definition and Methodology

3.1 Problem Definition

The problem definition of Algorithm Visualizer is to provide a user-friendly web-based application that enables users to understand how algorithms work by visualizing their execution step by step. Algorithms are a set of instructions that are executed to solve a particular problem. Understanding algorithms is essential in various fields, including computer science, engineering, mathematics, and data science.

However, algorithms can be challenging to understand, especially for beginners or non-technical users. Algorithm Visualizer addresses this problem by providing a platform that enables users to visualize the algorithms' execution. It makes it easier for users to understand the algorithm's logic and how it works by displaying the data structures, variables, and steps taken during its execution.

Algorithm Visualizer's primary goal is to simplify the learning process for users who want to understand algorithms, whether they are beginners or experts. It also provides a platform for teachers and students to demonstrate and learn algorithms visually. By providing a user-friendly interface, Algorithm Visualizer makes it easier for users to experiment with different algorithms, input data, and complexity levels.

Algorithm Visualizer also aims to provide a platform for researchers and developers to share their algorithms and implementations with the wider community. By including an algorithm library, Algorithm Visualizer enables users to explore and learn about various algorithms, including their descriptions, complexity levels, and implementations.

Overall, the problem definition of Algorithm Visualizer is to provide an easy-to-use and comprehensive platform for visualizing algorithms, enabling users to understand their logic, functionality, and execution.

3.2 Proposed Methodology

Here in the proposed system, the user can select whichever model or algorithm he/she wants to study. On its selection according to the algorithm, a graph or its visual representation will be generated. On starting the animation, a systematic and detailed animation will be shown so as to how the algorithm works for a better understanding. The animation speed can be controlled according to the user’s pace. After learning, the user can also test their knowledge by trying to predict the working before playing the animation.

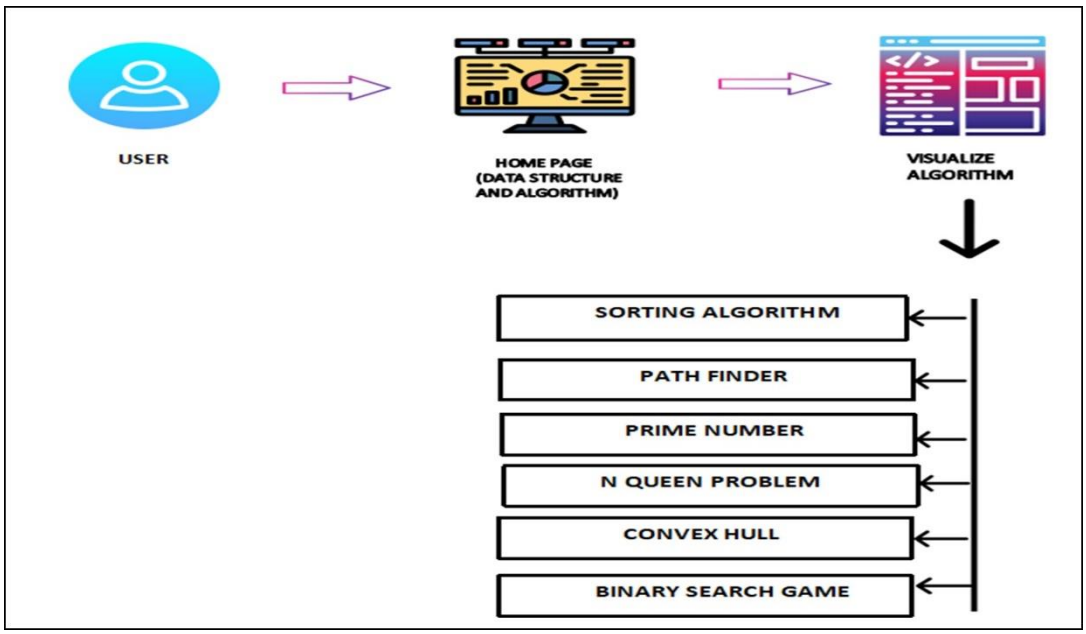


Fig. 3.1 System Architecture

Algorithm and process Design

1. GRAPH GENERATION ALGORITHM: In order for the system to provide a better educational experience and to give users the ability to test themselves on the same algorithm multiple times, it is necessary to use randomly generated graphs. It changes the values and randomly generates different graphs every time the user refreshes the tab.

2. BASIC DATA STRUCTURE ALGORITHMS: like linked lists, stacks, queues, sorting algorithms, searching algorithms, graph algorithms, geometric algorithms.

3. CPU Scheduling algorithms – There can be a number of processes running parallel in a system. So, the CPU (Central Processing Unit) must come up with a way by which it can choose from the waiting processes and complete their execution one by one. In doing so CPU must ensure the following things:

a) Maximum resource utilization

b) Minimum response time

c) CPU allocation must be fair enough

d) Maximum throughput

Considering the above criteria, the various popular scheduling algorithms are as follows:

a) First-Come, First-Served (FCFS) Scheduling

b) Shortest-Job-Next (SJN) Scheduling

c) Priority Scheduling

d) Round Robin (RR) Scheduling

4. Tree based visualization of algorithms: We have observed that in most of the standard books for algorithms, the working of algorithms is shown with reference to tree data structure. In order to ensure a gradual shift from books or classroom technique to e-learning, we can introduce this feature in our application and then in advanced levels we can introduce the existing, more efficient form of algorithm visualization for the students

Workflow

User can traverse to two different pages from the welcome page that are Path-finding Algorithm and Sorting Algorithm

PATH-FINDING ALGORITHM: The navbar of path-finding algorithm consists of the following options: -

Dijkstra's Algorithm: Depth First Search, Breadth First Search.

The algorithms present in the navbar are chosen on the basis of their popularity and difficulty level. Students find it difficult to understand these algorithms theoretically. When they will see the visualization of these algorithms then they will be able to understand it better. User will be able to differentiate between the functionalities of different algorithms on the basis of time complexity after the visualization is over.

MAZES AND PATTERNS: Maze and patterns are included to ensure better and clear understanding of algorithms. As there will be walls or obstruction between the starting node and the goal node, user can relate the visualization with real worldlike situation. Also, user will be able to figure out which algorithm is better based on algorithm time complexity. Especially for users looking for a playful option for understanding these complex topics this fun filled options can turn out to be the appropriate way

SPEED: The project contains speed bar for maintaining the speed of visualization, this feature is included because everyone has a different learning rate so the user can vary the speed of visualization according to his/her choice. Designing Grid structure will be used to represent each node. Computer generated starting and ending node will be displayed initially. User can change the positions of start and end node according to his/her will.

SORTING ALGORITHM: The navbar of sorting algorithm consists Merge Sort, Quick Sort, Heap Sort, Bubble Sort.

GENERATE NEW ARRAY: It will generate a new random array. Every time we click on this tab it will generate new random array. Array elements will be displayed in the form of bars with the height of each bar proportional to the numerical value it is representing. While sorting different coloured bars would be used to represent the sorted, unsorted and currently sorting numerical values from the array of input numbers.

CHANGE ARRAY SIZE AND SORTING SPEED: A slider will be provided so that user can change the size of array and accordingly the speed of sorting will vary. Size of array will be directly proportional to the sorting speed i.e. (larger the speed of array greater will be the speed of sorting). As, mentioned earlier this feature has been implemented to ensure users are able to learn at their own pace without any haste.

Algorithm and process Design

1. GRAPH GENERATION ALGORITHM: In order for the system to provide a better educational experience and to give users the ability to test themselves on the same algorithm multiple times, it is necessary to use randomly generated graphs. It changes the values and randomly generates different graphs every time the user refreshes the tab.

2. BASIC DATA STRUCTURE ALGORITHMS: like arrays, linked lists, stacks, queues, sorting algorithms, searching algorithms, graph algorithms, geometric algorithms.

N-QUEEN PROBLEM: The N Queen is the problem of placing **N** chess queens on an **N×N** chessboard so that no two queens attack each other. For example, the following is a solution for the 4 Queen problem.

The N-Queens problem can be solved using various algorithms and techniques, including backtracking, brute force, and various heuristics. The problem has practical applications in computer science, such as in cryptography, game theory, and optimization. Additionally, the N-Queens problem has been used as a benchmark for evaluating the performance of algorithms and parallel computing architectures.

TURING MACHINE: A Turing machine is a mathematical model that was proposed by Alan Turing in 1936. It is a theoretical machine that can simulate any algorithmic computation, making it a powerful tool for studying the limits of computation.

Our Turing machine is designed to perform three tasks:

I. Bitwise Not

II. Add one

III. 2's complement

Bitwise not is a simple operation that flips the bits of a binary number, while add one and 2's complement are used to perform arithmetic operations on binary numbers.

CONVEX HULL: Convex hull is a concept in geometry and computational geometry. It refers to the smallest convex polygon that encloses a given set of points in a two-dimensional plane or a higher-dimensional space. In simpler terms, it is the boundary of the set of points that is the outermost and forms a convex shape.

The convex hull can be visualized as a rubber band that is stretched around a set of points, such that the band lies tightly around the points and does not cross itself. All the points on the rubber band are part of the convex hull.

The concept of convex hull has various applications in areas such as computer graphics, pattern recognition, image processing, and geographical information systems.

3.3 System Design

The system design of Algorithm Visualizer involves several components working together to provide a seamless user experience. Here is an overview of the different components:

Front-end: The front-end of the system is responsible for rendering the user interface and displaying the visualizations of the algorithms. It is built using HTML, CSS, and JavaScript frameworks such as React.

Back-end: The back-end of the system handles the logic and processes the data entered by the user. It is built using a server-side language such as Node.js or Python and is responsible for handling user requests, performing computations, and generating visualizations.

Overall, the system design of Algorithm Visualizer is focused on providing an intuitive and interactive user experience while leveraging powerful algorithms to help users learn and understand complex concepts.

3.3.1 System Model

The proposed system involves the simulation of the different type of algorithms codes. As you can see, there are no major components besides the three coding languages. Most websites have tools or scripts that require a server on the back-end (like PHP), but it is not necessary in this case since React JS runs right in the user's browser.

JSX and CSS are used for the interface. The JSX communicates with the React JS code and vice versa to launch the appropriate algorithms and update the interface accordingly, as seen with a single, bidirectional arrow. As the React JS was modified from a functional programming focus to a more object-oriented one, the parts of the JSX that did change were the function calls for each button. All of the back-end interaction is abstracted to the various buttons for selecting algorithms and running the animation.

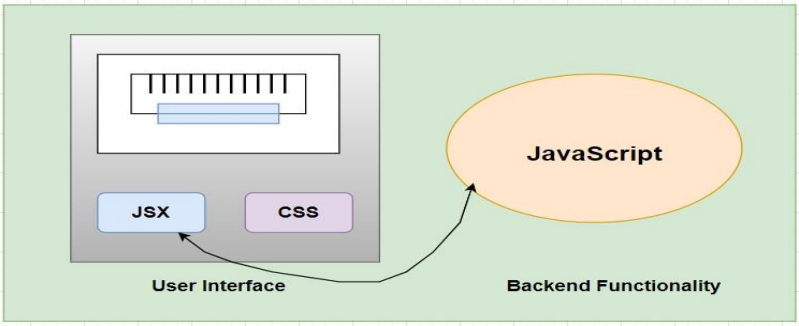


Fig. 3.2 System Model

3.3.2 ER Diagram

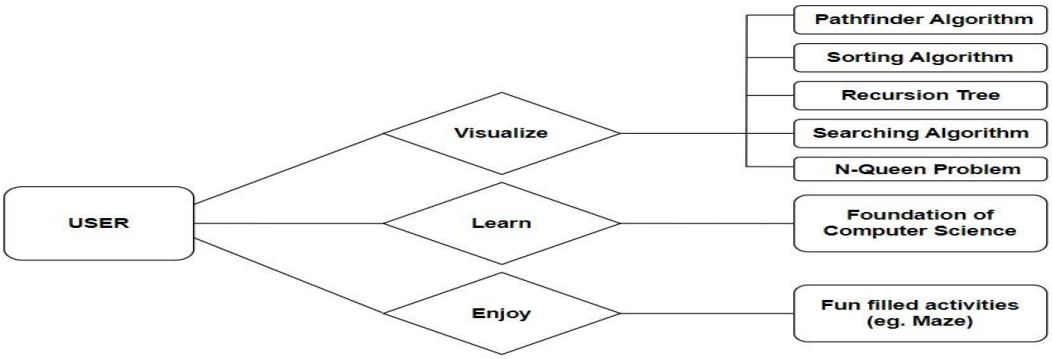


Fig. 3.3 ER Diagram

3.3.3 DFD

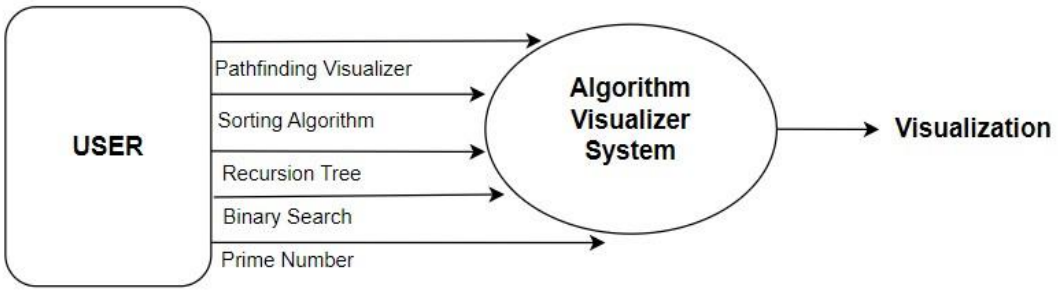


Fig. 3.4 DFD 0

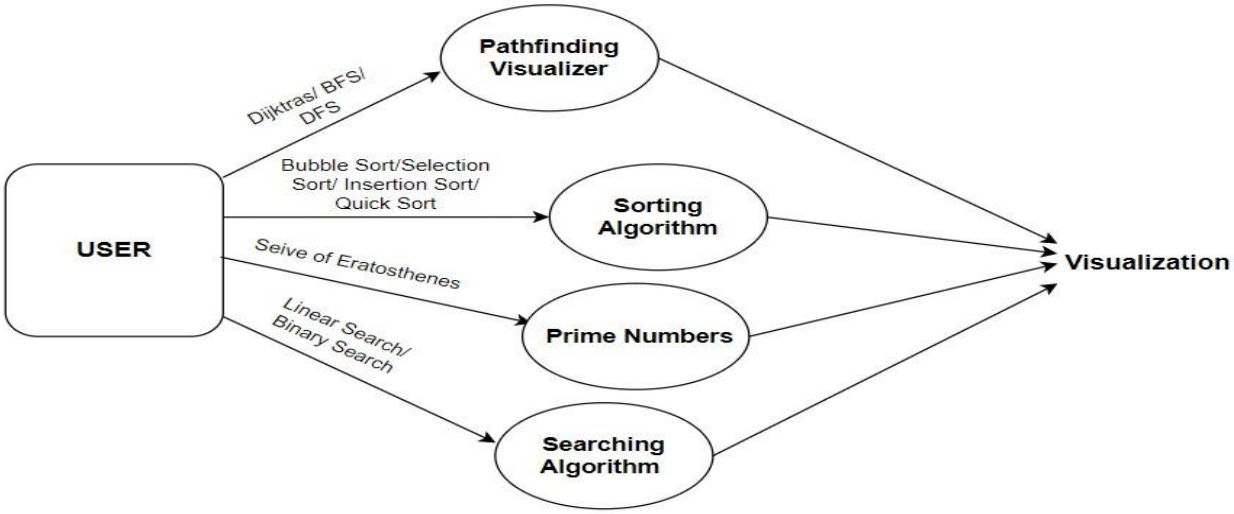


Fig. 3.5 DFD 1

3.3.4 Data Dictionary

Algorithms: A list of algorithms that Algorithm Visualizer supports, including their names and descriptions.

Input Data: The data provided by the user or selected from pre-defined test cases, including the data type, format, and size.

Animation: The step-by-step animation generated by Algorithm Visualizer that displays the execution of the algorithm, including the animation style, speed, and controls.

Output: The output generated by the algorithm, including its data type, format, and size.

Time Complexity: The time taken by the algorithm to execute on the input data, including the time complexity notation and the number of operations performed.

Space Complexity: The space required by the algorithm to execute on the input data, including the space complexity notation and the amount of memory used.

User Interactions: The actions performed by the user to interact with the visualization, including pause, play, rewind, and fast-forward.

User Interface: The visual and interactive elements of Algorithm Visualizer, including the design, layout, and features.

Performance Metrics: The performance metrics used to evaluate the efficiency and effectiveness of Algorithm Visualizer, including user feedback, usage statistics, and system logs.

Overall, the data dictionary of Algorithm Visualizer would include information on the algorithms supported, input and output data formats, animation style and controls, time and space complexity, user interactions, user interface, and performance metrics.

3.3.5 Flow Chart

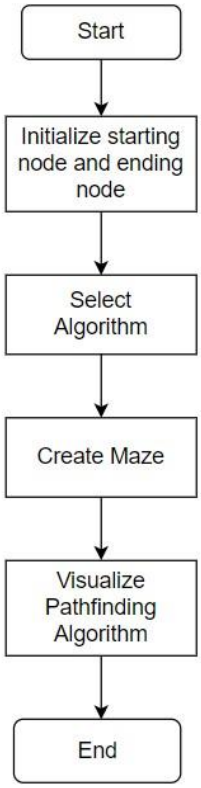


Fig. 3.6 Pathfinding Visualizer

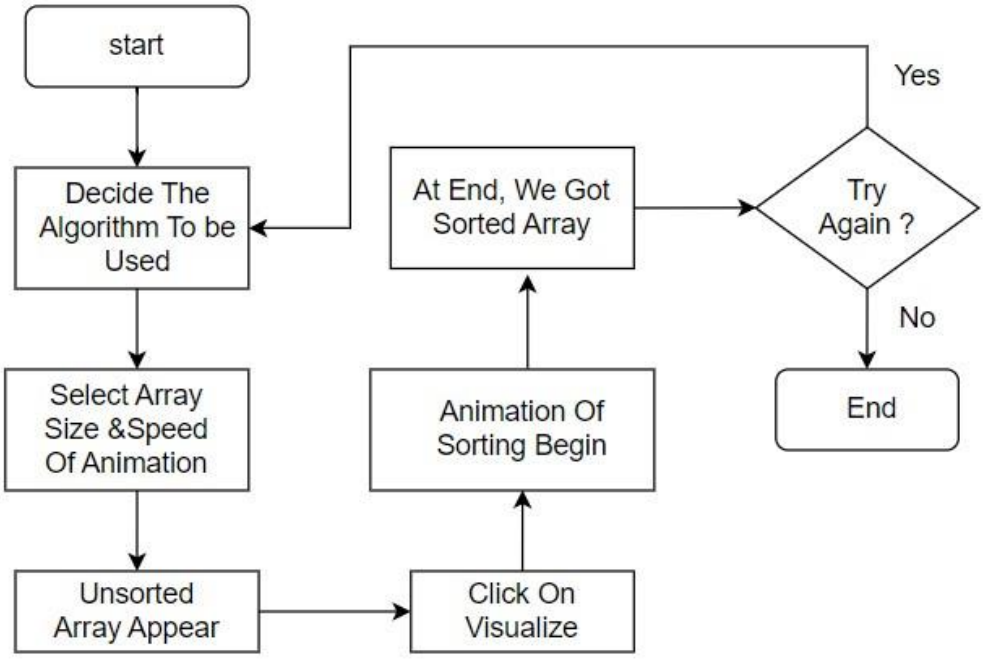


Fig. 3.7 Sorting Visualizer

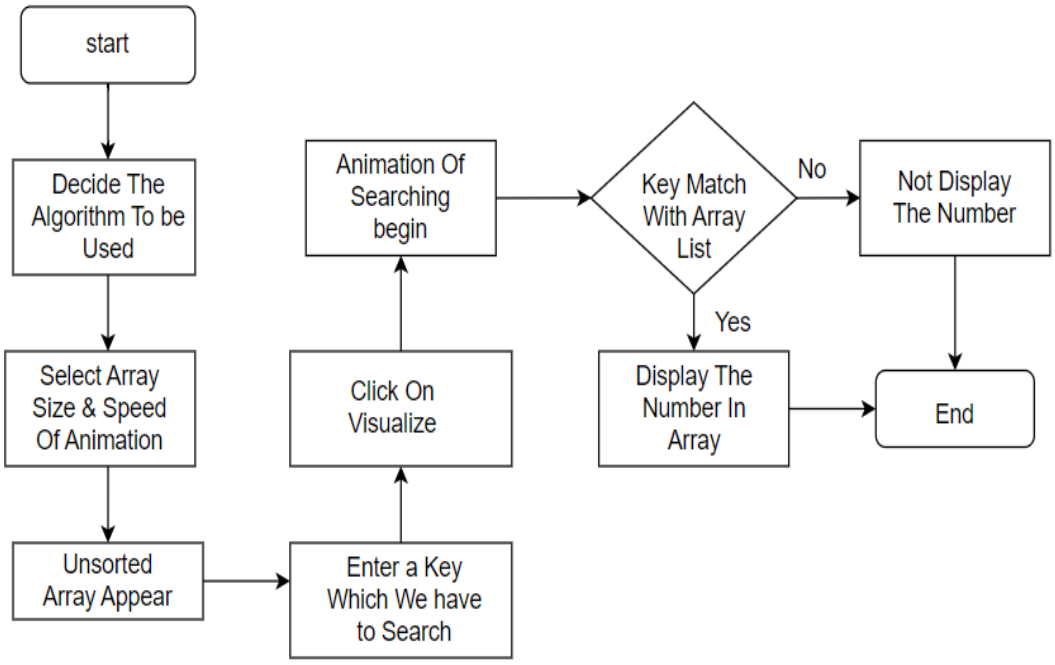


Fig. 3.8 Searching Visualizer

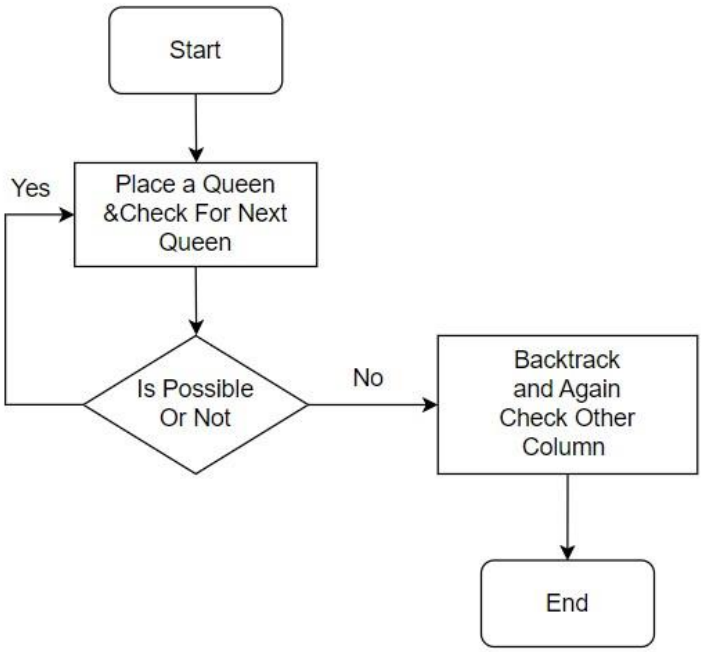
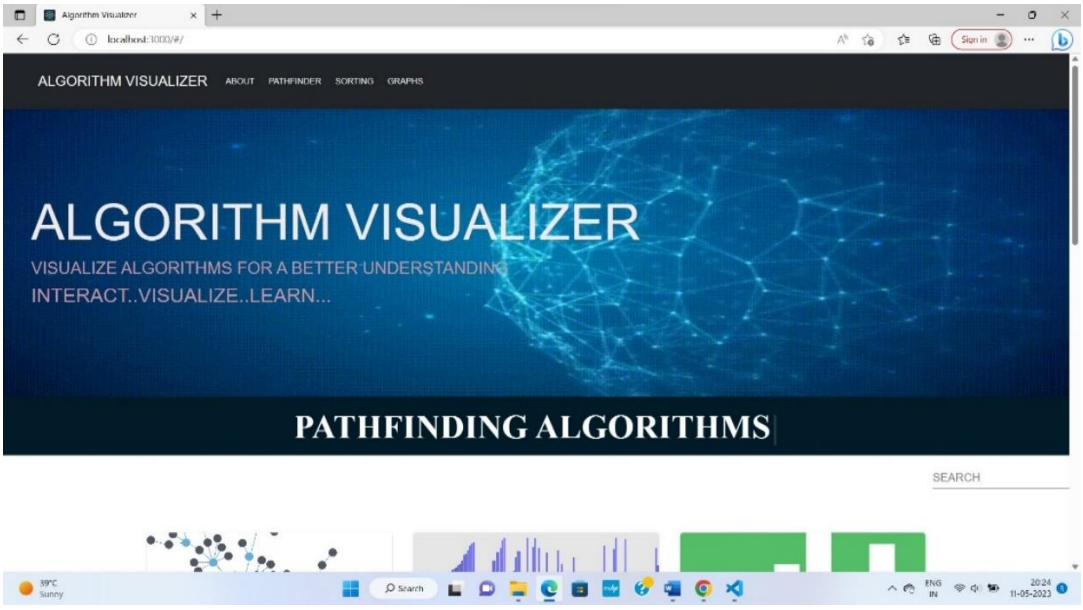


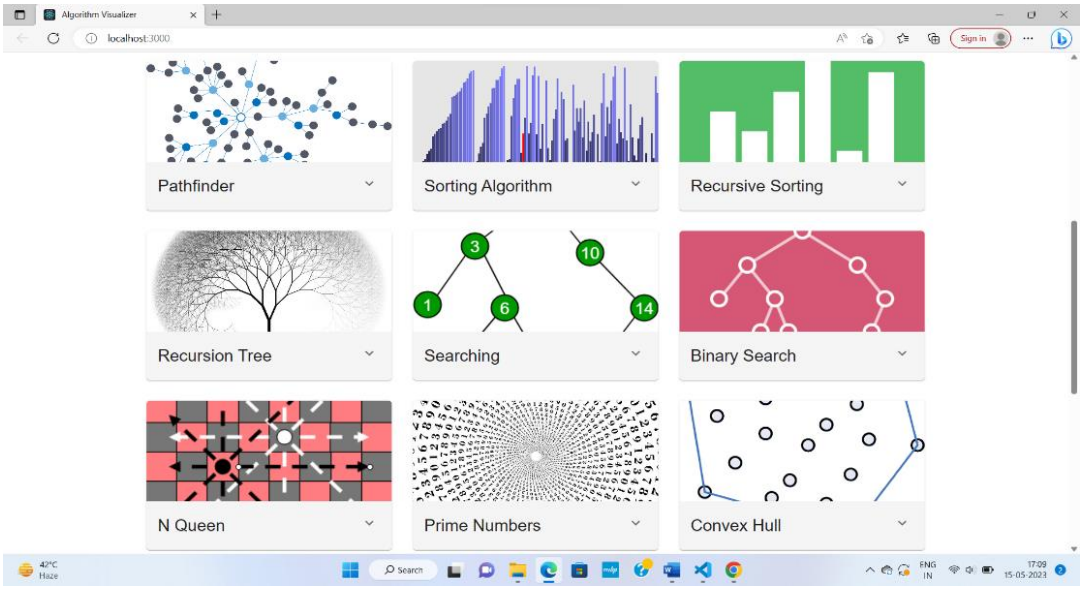
Fig. 3.9 N-Queen Problem

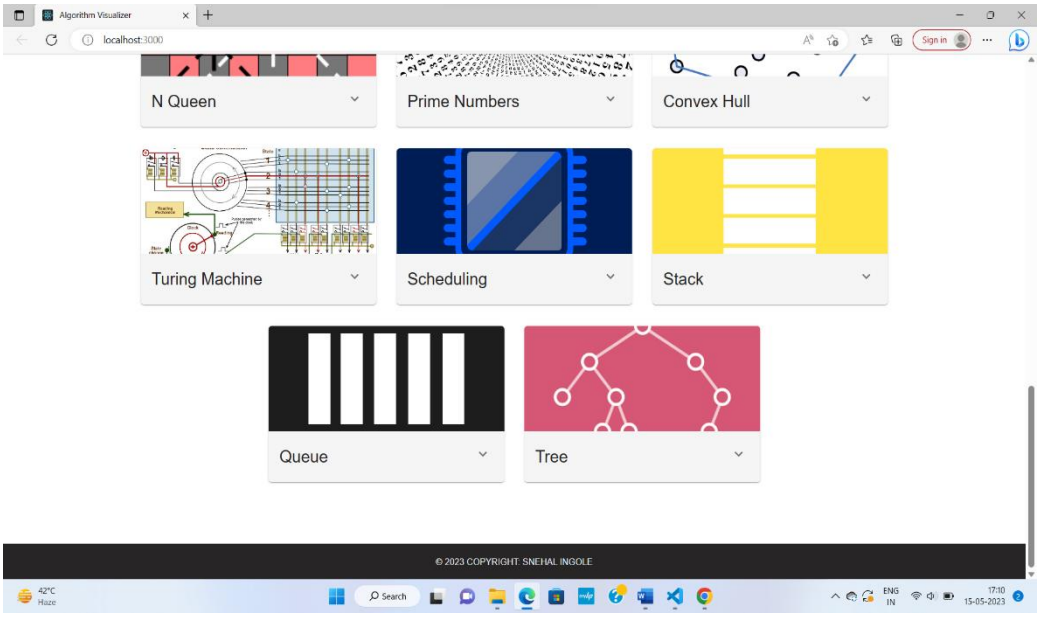
Chapter 4 Implementation

User interface / Front end

- **Home Page**



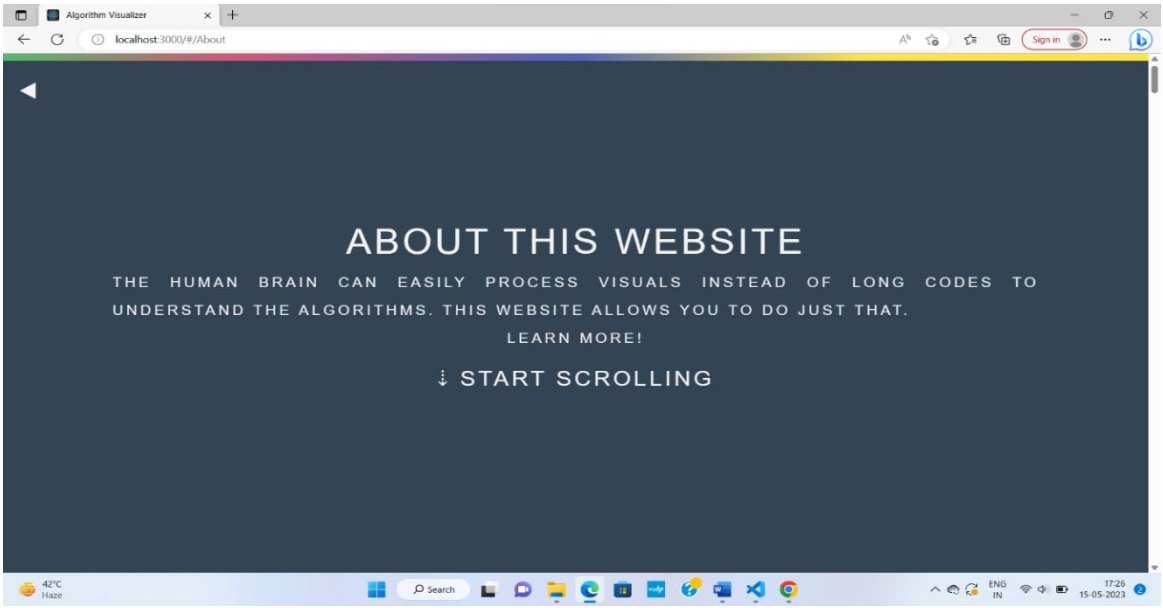




In our Algorithm visualization project this is the home page of our project. The main goal of our project is to provide the visual representation of various algorithms to understand the algorithms easily.

In our home page various options of algorithms for visualization such as pathfinder, sorting, recursive sorting, recursion tree, n-queen, turing machine, prime number, scheduling, stack, queue, convex hull, binary search tree, searching algorithms are present. Also it contains the search option to search the specific algorithm easily.

About Page



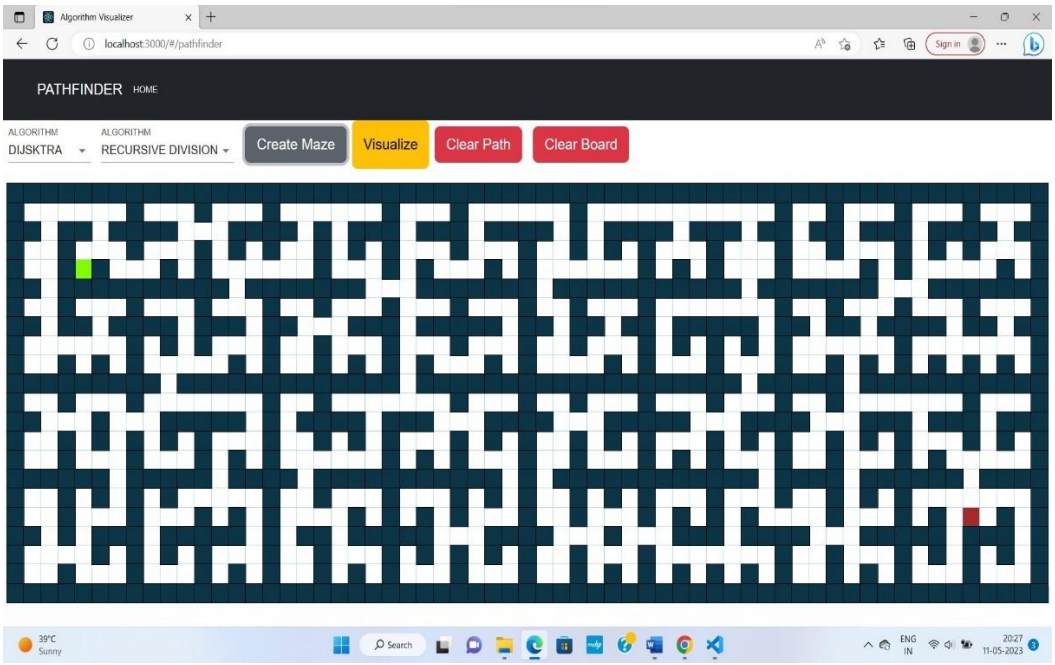
About section contains information about the algorithms that is implemented in the algorithm visualization project.

Chapter 5 Result and Discussion

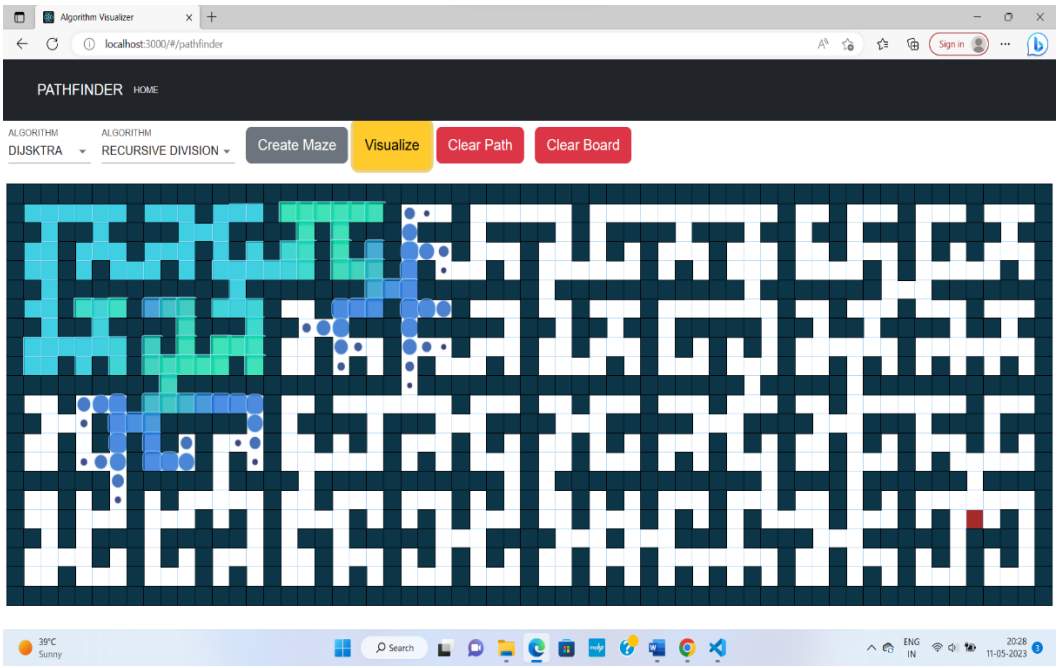
5.1 Result

For Pathfinding visualizer

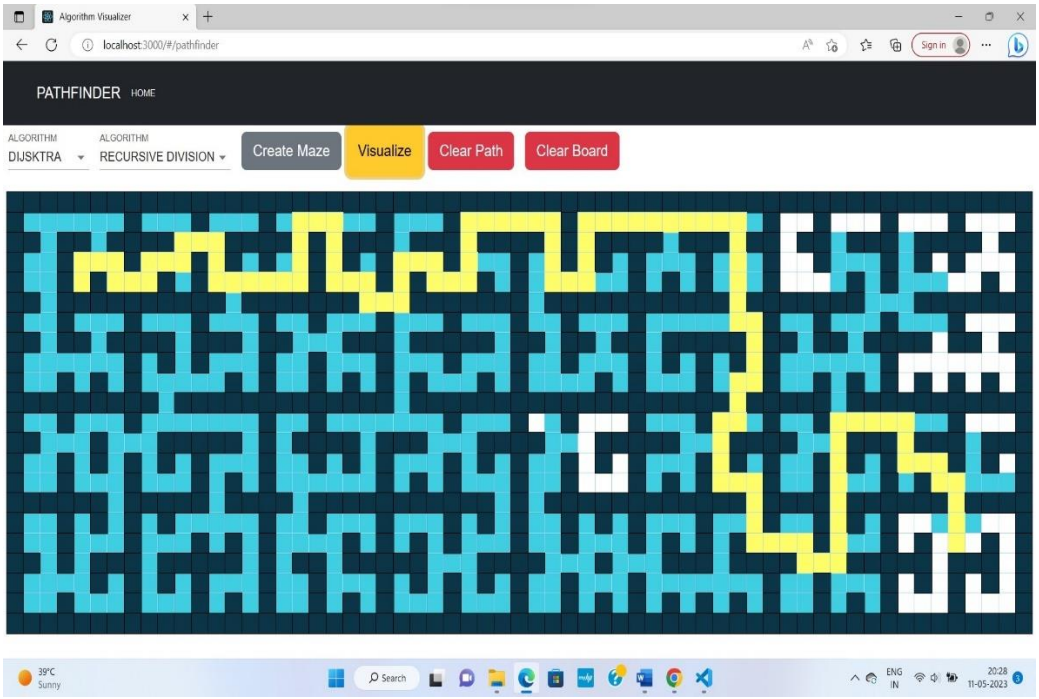
Before Visualization:



While Visualizing:

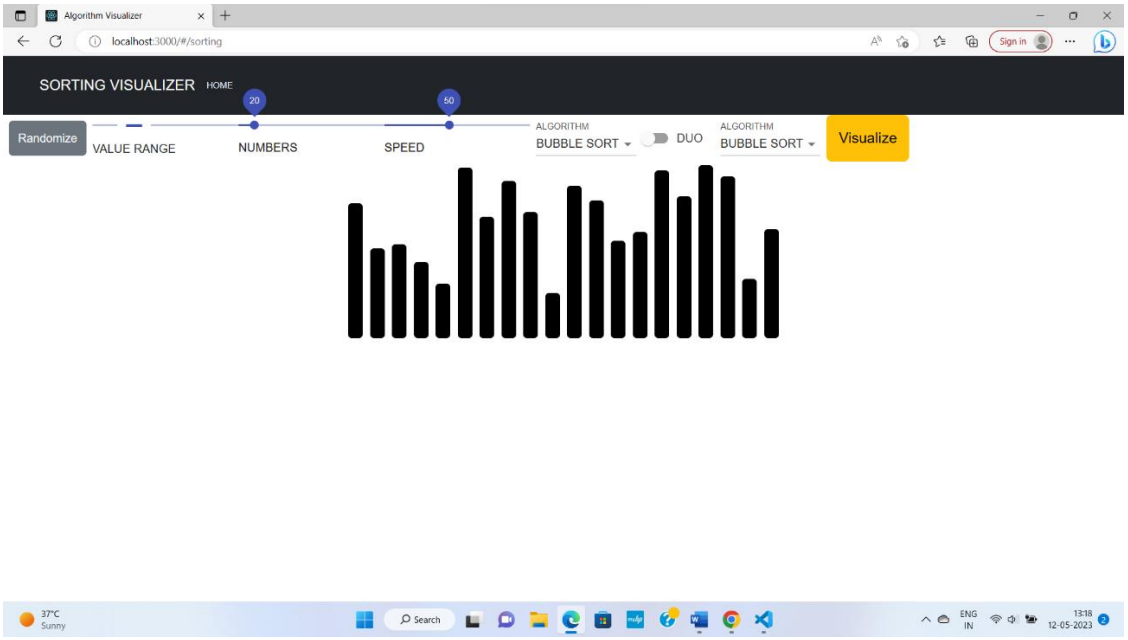


After Visualizing:

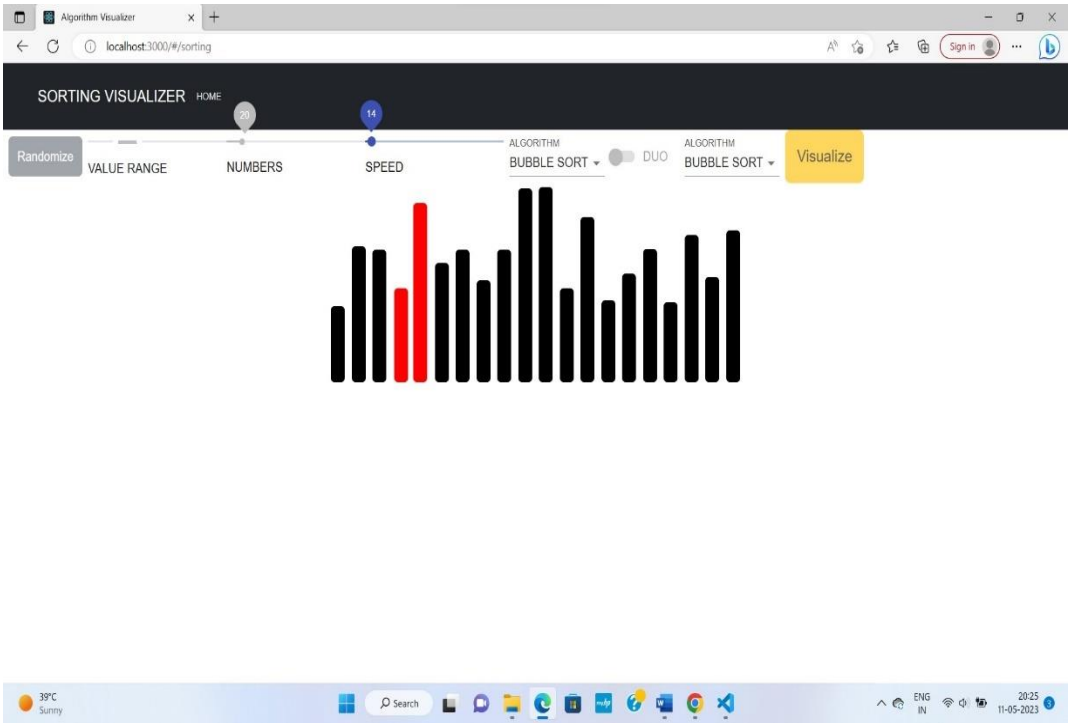


For Sorting Visualizer:

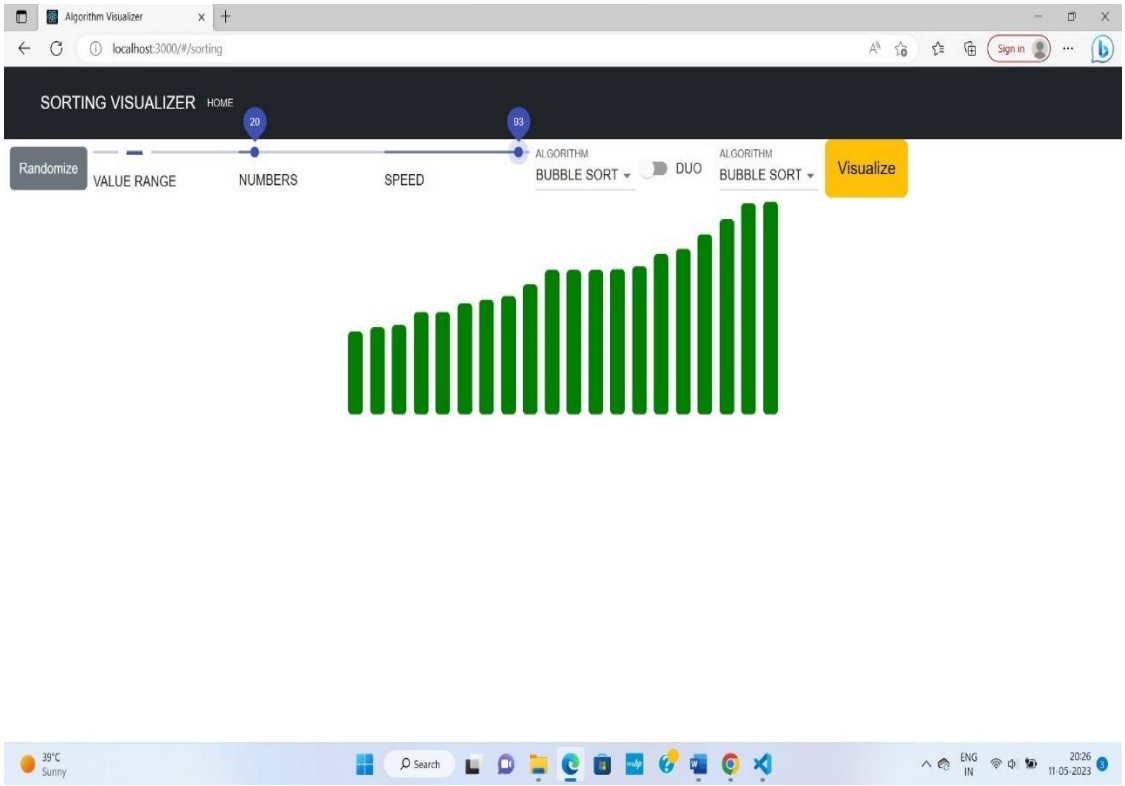
Before Visualization:



While Visualizing:

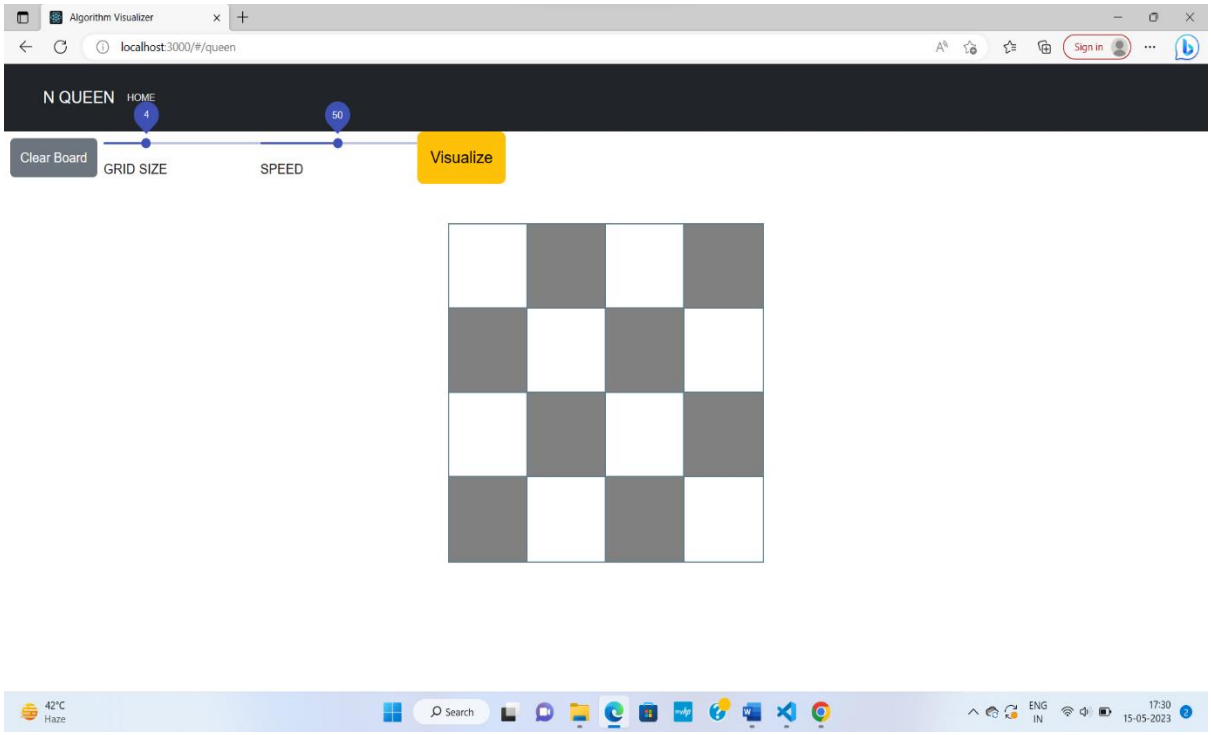


After Visualization:

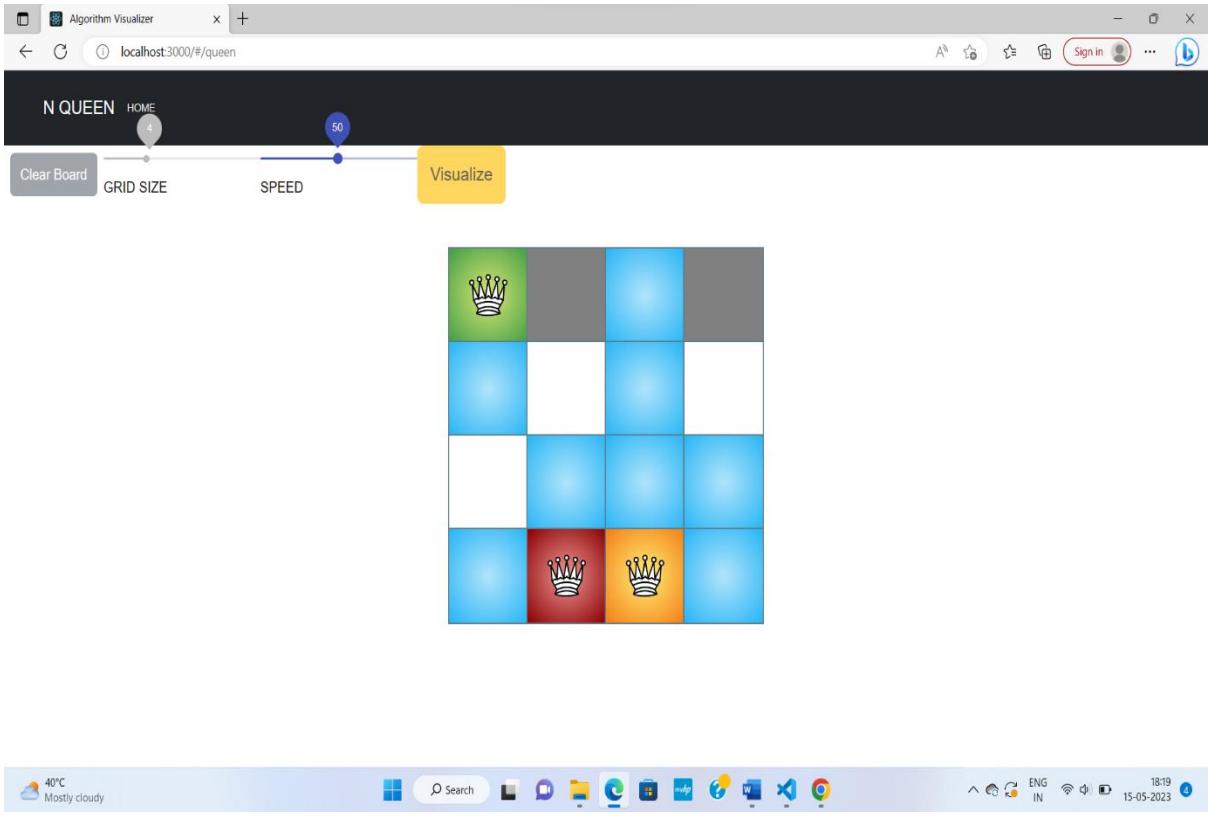


For N-Queen Problem

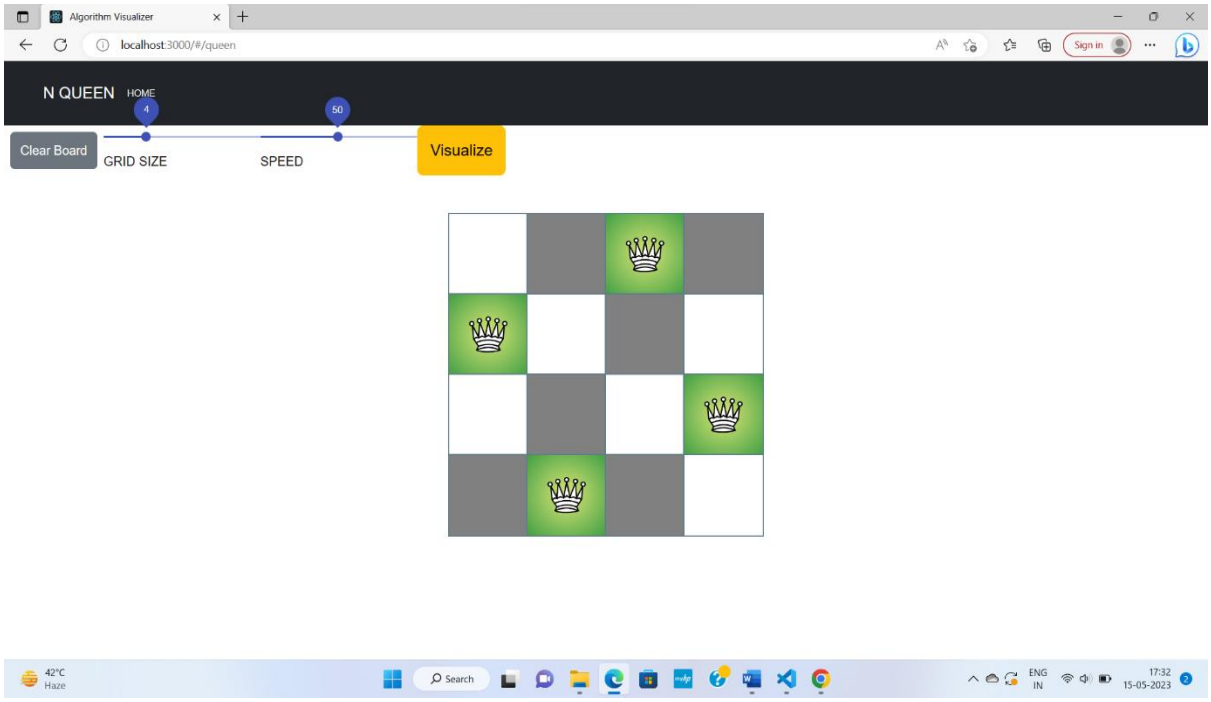
Before Visualization:



While Visualizing:

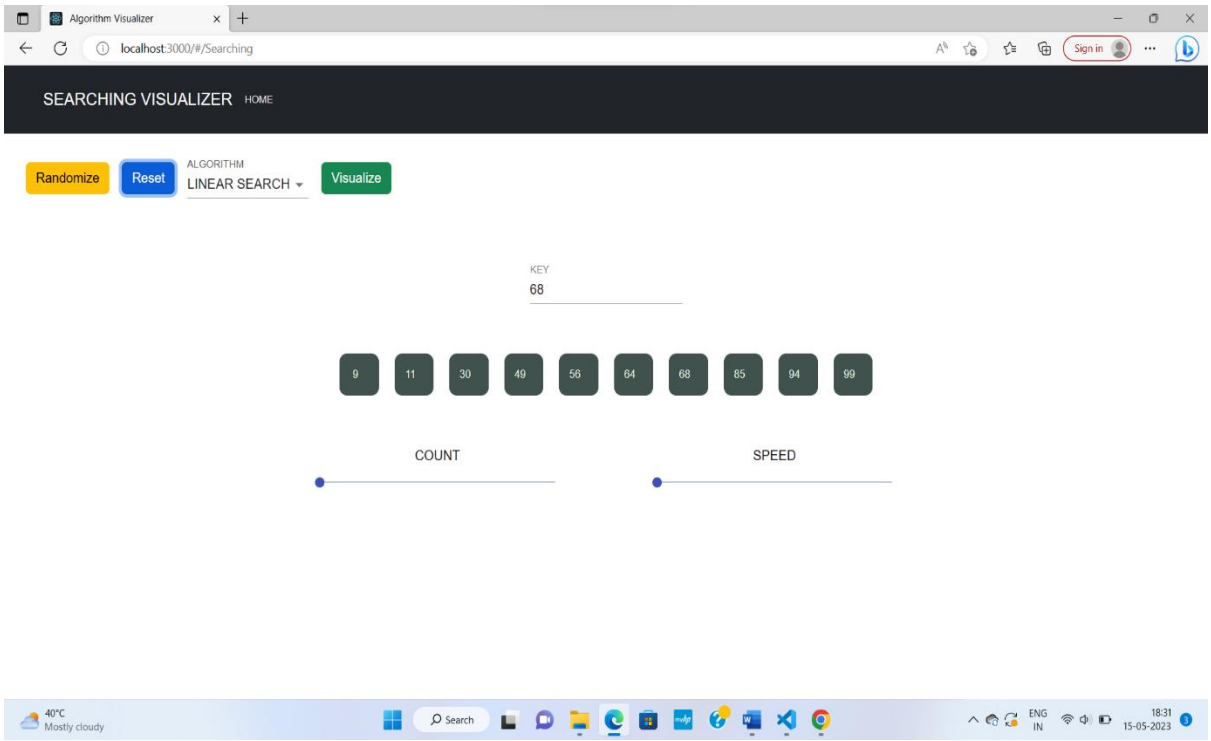


After Visualization:

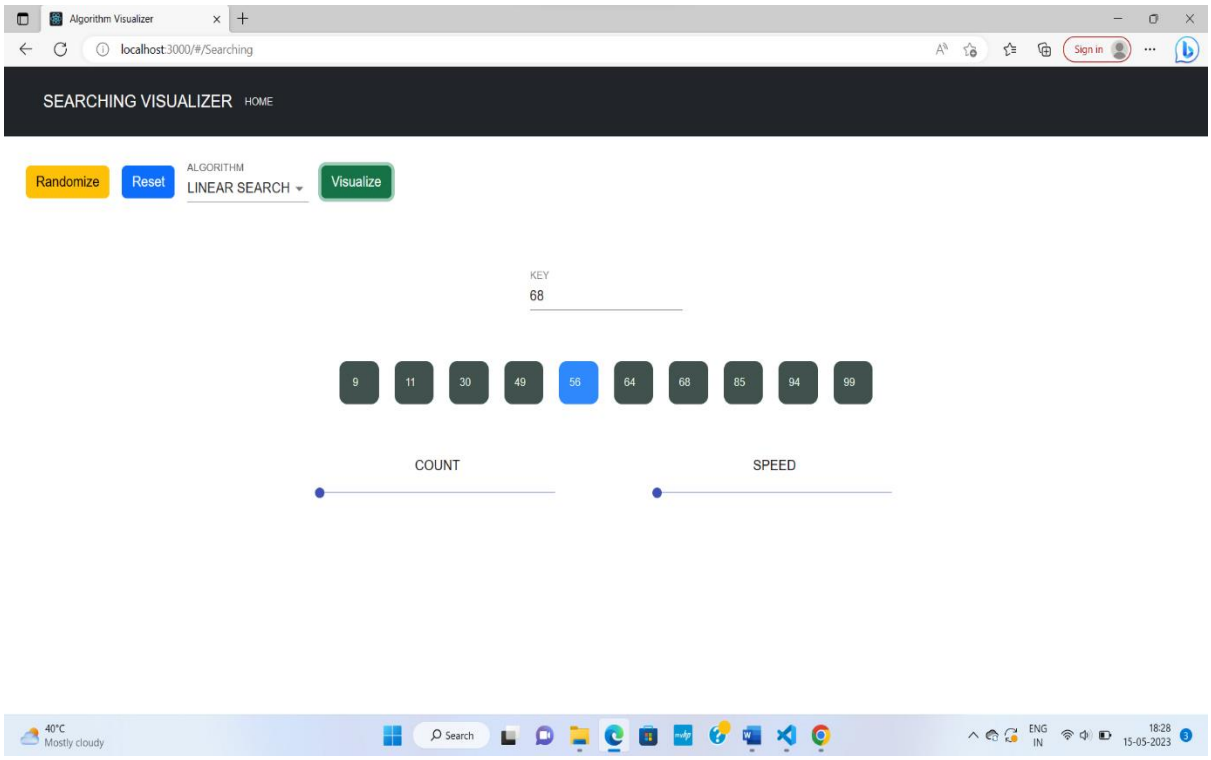


For Searching Algorithm:

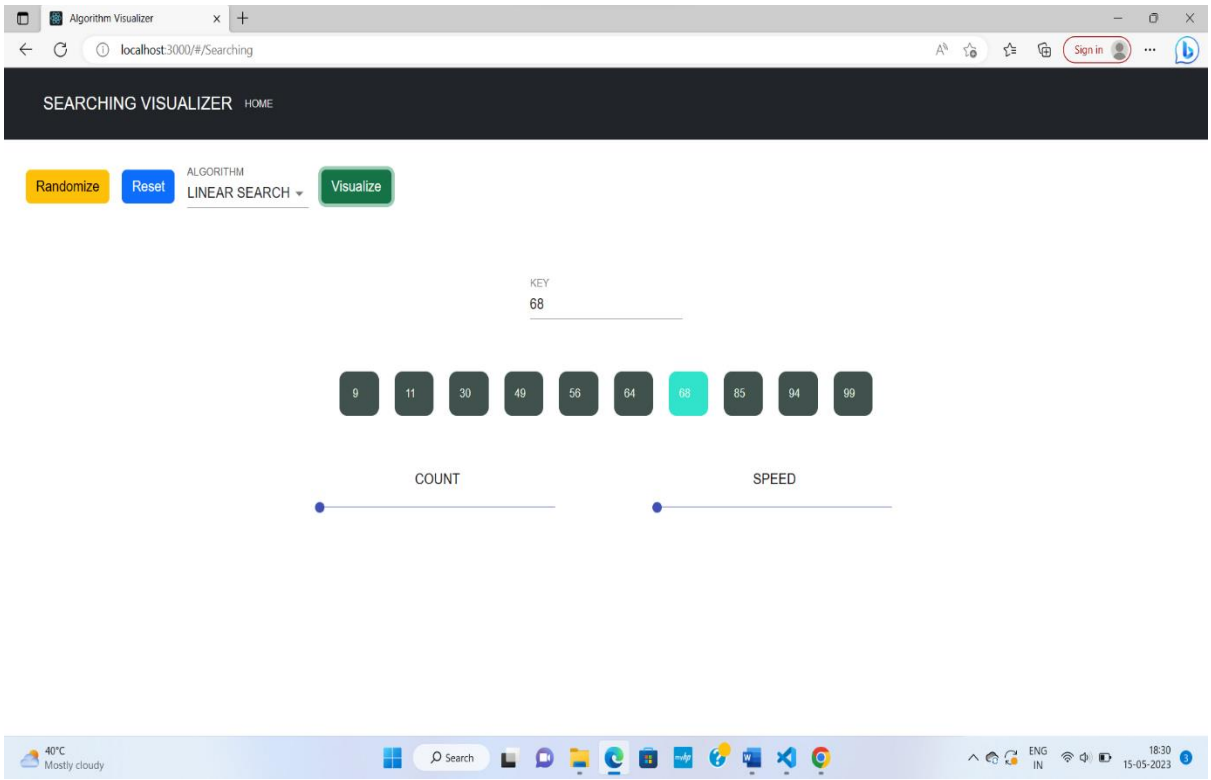
Before Visualization:



While Visualizing:

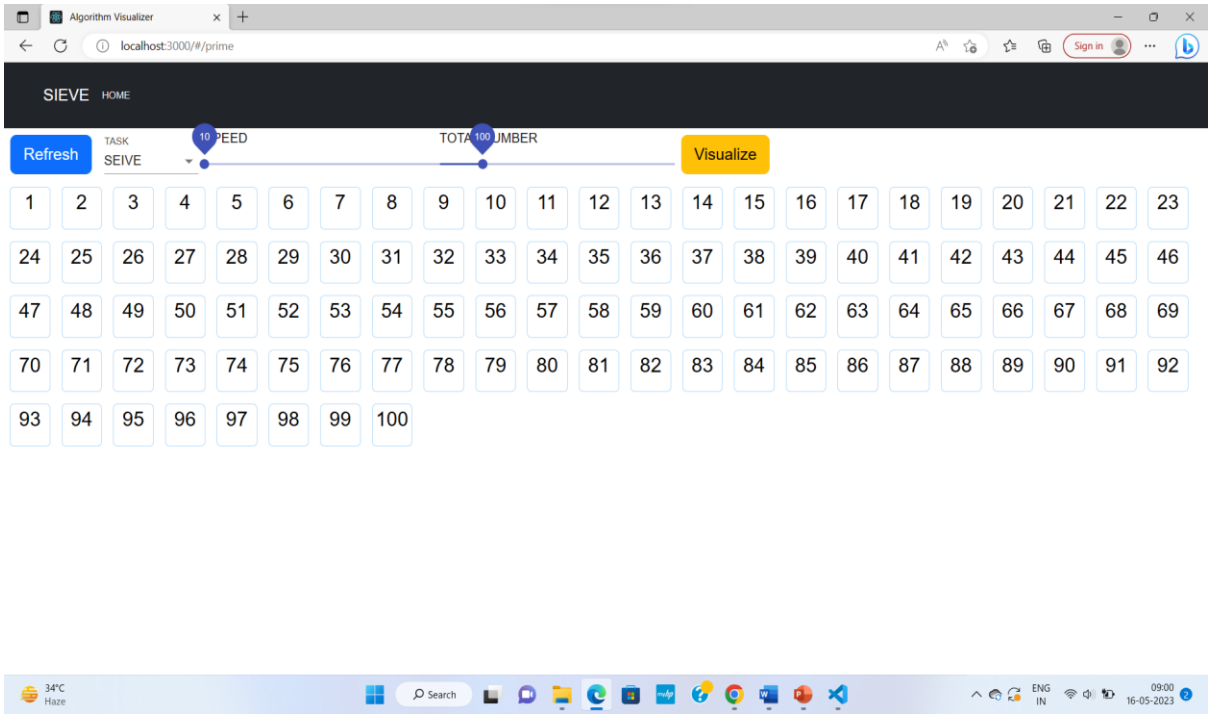


After Visualization:

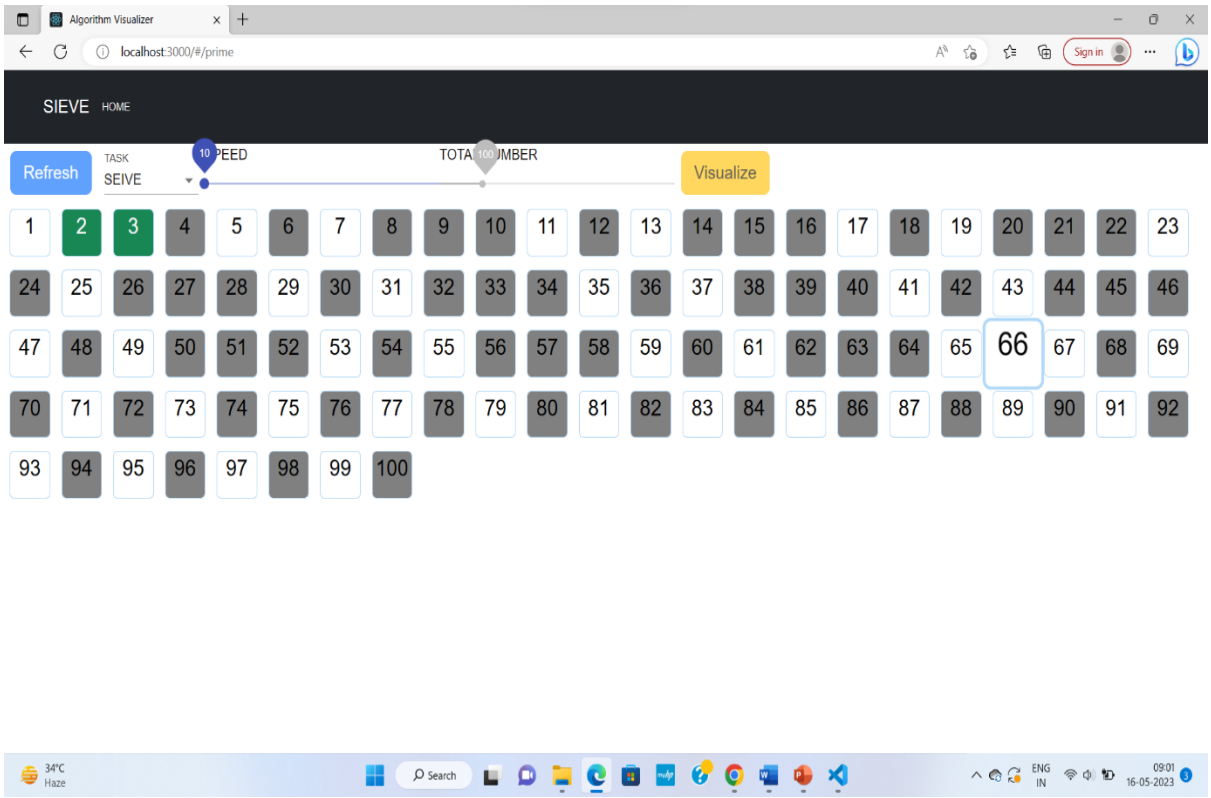


For Prime Number:

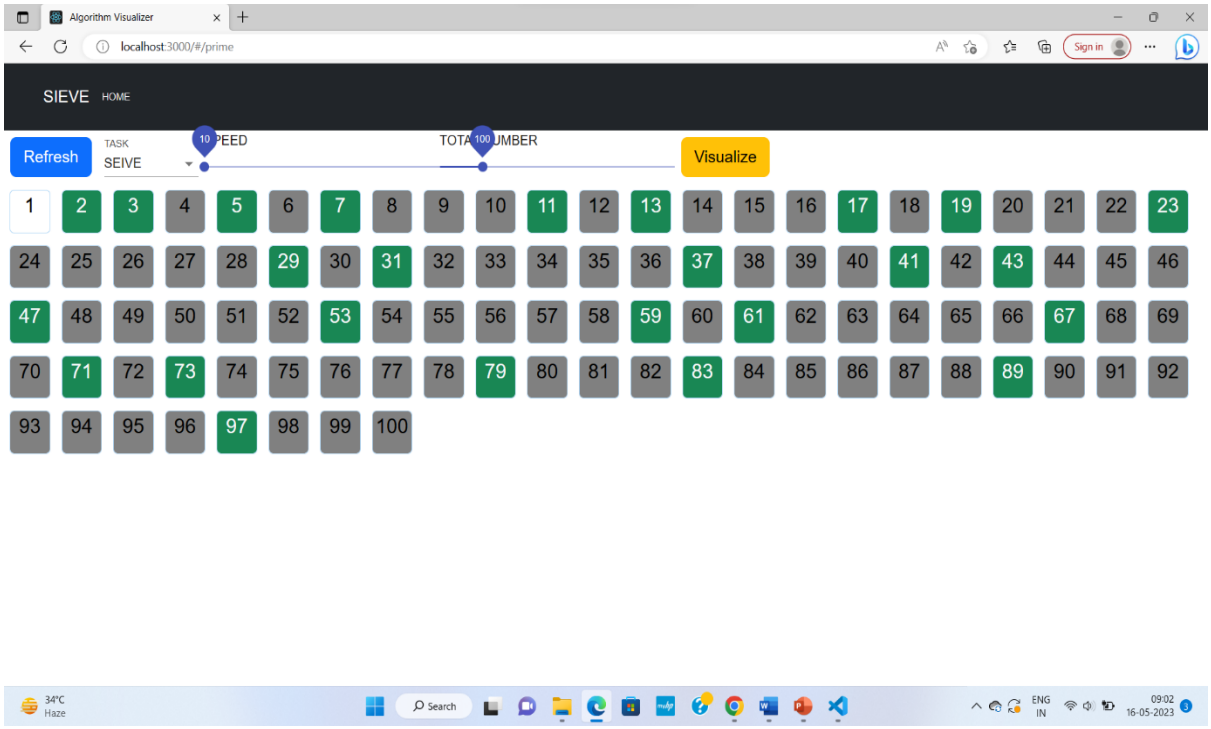
Before Visualization:



While Visualizing:



After Visualization:



5.2 Discussion

Algorithm visualizer has been widely praised for its ability to help learners understand complex algorithms by providing an interactive and visual experience. Many users have found it to be an engaging and effective way to learn and practice algorithms, and it has been featured in several educational resources and programming courses.

One of the main benefits of algorithm visualizer is that it allows users to see how an algorithm works step by step, which can be difficult to understand by just reading the code. The visualizations are clear and intuitive, making it easy to follow the algorithm's logic and understand how it solves a particular problem.

Another advantage of algorithm visualizer is that it provides an opportunity for learners to experiment with different inputs and parameters, allowing them to see how the algorithm behaves under different conditions. This can help learners develop their problem-solving skills and gain a deeper understanding of the algorithm.

There are also some potential drawbacks to algorithm visualizer. For example, some users may become overly reliant on the visualizations and not develop their ability to read

and write code. Additionally, some visualizations may oversimplify or omit certain aspects of the algorithm, leading to a less accurate understanding.

Overall, algorithm visualizer is a valuable tool for learners and educators in computer science and related fields. By combining interactive visualizations with algorithmic concepts, algorithm visualizer provides an accessible and effective way for users to learn and practice algorithms.

Chapter 6 Conclusion and Future Scope

6.1 Conclusion

We started our project by studying a number of the well-known algorithm visualizations that are developed over a few years. According to our findings, algorithmic visualization are often seen as a valuable supporting tool, utilized in addition to straightforward ways of education within the field of computer science.

With the execution of this project, we have got with success attain our objective of our project is to engraft Graph Path Finding with Visualization and differentiate their performance. As is the case with most other teaching areas, there has been a major gap between the idea and practical understanding of algorithms realization. This is often true for shortest paths algorithms and in specially for Dijkstra algorithm. The main goal of the project is to use it from research educators and students for teaching and studying the existing known graph algorithms. The main plan of the system is to provide an associate educational environment for both instructors and students to facilitate the learning process in economical way.

6.2 Future Scope

The future scope of algorithm visualizer is quite promising, as it has the potential to become an essential tool for learning and teaching computer science. Here are a few possible areas of improvement and expansion for algorithm visualizer:

Adding more algorithms: Algorithm visualizer can expand its library of algorithms to cover more topics, such as machine learning algorithms, network algorithms, and cryptography algorithms

Customization options: Users may benefit from more customization options, such as the ability to change the visualization style, modify the speed of the animation, and adjust the input size and values.

Collaboration and sharing features: Algorithm visualizer can add features that allow users to collaborate on algorithms, share visualizations, and comment and discuss the execution process.

Integration with education platforms: Algorithm visualizer can integrate with popular education platforms, such as Coursera, edX, and Khan Academy, to provide learners with a seamless learning experience.

Interactive tutorials: Algorithm visualizer can develop interactive tutorials that guide learners through the execution process, provide feedback, and offer hints and tips to help them master the algorithm.

Overall, the future scope of algorithm visualizer is vast, and it has the potential to enhance the way people learn and teach computer science.

References

[1] https://www.irjmets.com/uploadedfiles/paper//issue_5_may_2022/22285/final/fin_irjmets1652877275

[2] https://ieeexplore.ieee.org/document/9667586

[3] https://ijcrt.org/papers/IJCRT2204656

[4] https://algorithm-visualizer.org/

[5] https://www.studocu.com/in/document/chandigarh-college-of-engineering-and-technology/algorithm-visualizer/algorithm-visualizer-synopsis/41156935

[6] https://theses.cz/id/3w4s3a/Mykhailo-Klunko-bc-thesis.pdf

[7] https://kkshitiz.github.io/Algorithm-Visualizer/

[8] https://easychair.org/publications/preprint_download/tw6K

[9] https://link.springer.com/10.1007/978-0-387-30162-4_464

[10] https://www.researchgate.net/publication/308187189_Critical_Analysis_on_Algorithm_Visualization_Study

[11] https://www.irjmets.com/uploadedfiles/paper//issue_5_may_2022/22285/final/fin_irjmets1652877275.pdf

[12] https://www.studocu.com/in/document/chandigarh-college-of-engineering-and-technology/algorithm-visualizer/algorithm-visualizer-synopsis/41156935