SORTING VISUALIZER

A Mini Project-II Report submitted
in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

1. 19B01A05F3 - P.Meghana Sri Manvitha
2. 19B01A05G8 - T.Aswitha
3. 19B01A05H9 - V.Nikitha Devi
4. 19B01A05J0 - Y.Leena Prasanna
5. 20B05A0518 - T.Sai Vineela

Under the esteemed guidance of

Mr. G. V. S. S. Prasad Raju M.Tech
Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)
(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)

BHIMAVARAM – 534 202
2021 – 2022

SHRI VISHNU ENGINEERING COLLEGE FOR WOMEN(A)

(Approved by AICTE, Accredited by NBA & NAAC, Affiliated to JNTU Kakinada)

BHIMAVARAM – 534 202

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Mini Project-II entitled “ ***Sorting Visualizer*** *”, is being submitted by* ***P.Meghana Sri Manvitha, T.Aswitha, V.Nikitha Devi, Y. Leena Prasanna, T.Sai Vineela*** *bearing the* ***Regd. No. 19B01A05F3, 19B01A05G8, 19B01A05H9, 19B01A05J0,20B05A0518*** *in partial fulfillment of the requirements for the award of the degree of “****Bachelor of Technology*** *in* ***Computer Science & Engineering****” is a record of bonafide work carried out by them under my guidance and supervision during the academic year* ***2021 – 2022*** *and it has been found worthy of acceptance according to the requirements of the university.*

Internal Guide **Head of the Department**

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task wouldbe impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing my deep sense of gratitude to **Dr. K. V. Vishnu Raju**, Chairman of Shri Vishnu Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. G. Srinivasa Rao**, Principal and **Dr. P. Srinivasa Raju**, Vice-principal of SVECW for their constant support and encouragement.

I would also like to thank **Dr. P. Kiran Sree**, Professor and Head of the Department of Computer Science and Engineering, for his constant support.

I am grateful to **Dr. P. Kiran Sree (HOD), Mr. P. Raju (Project Co-Ordinator)** and **Mr. G. V. S. S. Prasad Raju (Project Guide) - PRC Members** for their unfailing encouragement and suggestions, given to me in the course of my project work.

I express my gratitude to **Mr. G. V. S. S. Prasad Raju, Assistant Professor**, my project guide, forconstantly monitoring the development of the project and setting up precise deadlines. Hisvaluable suggestions were the motivating factors in completing the work.

Finally, a note of thanks to the teaching and non-teaching staff of Dept of Computer Scienceand Engineering, for their cooperation extended to me, and my friends, who helped me directly or indirectly in the course of the project work.

Project Associates :

19B01A05F3 - P.Meghana Sri Manvitha
19B01A05G8 - T.Aswitha
19B01A05H9 - V.Nikitha Devi
19B01A05J0 - Y.Leena Prasanna
20B05A0518 - T.Sai Vineela

ABSTRACT

This project outlines a study that tested the benefits of animated sorting algorithms for teaching a algorithm based Visualization. To visualize four sorting algorithms, a web-based animation application was constructed. A visualization of data is implemented as a bar graph, after which a data sorting and algorithm may be applied. The resulting animation is then performed either automatically or by the user, who then sets their own pace and input array size, speed of the visualization.

This is research on the computer science curriculum's approach to learning sorting algorithms which are categorized into quadratic and logarithmic like bubble, insertion, selection, quick, merge and heap sorting techniques etc., This project features the importance of each algorithm individually using it’s time and space complexities. The experiment featured a presentation and a survey, both of which asked students questions which may illustrate improvements in algorithm comprehension. These findings and reactions are catalogued in this document and compared to earlier investigation

Keywords: Sorting Algorithms, React Visualizer, Selection Sort, Merge Sort, Bubble Sort, Insertion Sort, Heap Sort.

Contents

INTRODUCTION

1.INTRODUCTION

How do you get something done? You don't have to be extremely complex in solving the problem, for example, if your car's headlight is broken .The main issue is figuring out the best way to go about it. To locate step-by-step directions in your car's handbook, you conduct research, or do you use instinct to find someone who knows how to do it? In short, our instinct tells me that we are a visual learners and hence more suited to acquire topics by watching them than by reading about them. In this case, we found that seeing the data move to its rightful spot as the result of an algorithm is much easier to follow than looking at the source code and trying to figure out where the data was supposed to go. This project was born out of curiosity about sorting algorithms, which inspired the idea for this project, which details an online tool we built that explains how sorting algorithms transform and organize sets of data. It is possible to organize a list of people, for example, by their age in ascending order using different methods

Each number is depicted as a bar, and the height of each bar represents the value of that number. It is being shifted by the algorithm from its original, unordered location to its final ordered place, making it distinct from the rest of the data. Selection Sort, Bubble Sort, Insertion Sort, and Merge Sort are the four sorting algorithms.

We wanted the animation to appeal to a wide spectrum of individuals utilizing various technology media, and so we had it made in a web-based format. Instead of requiring the user to install extra software or attempt to organize setups to use the tool, this helps to remove this source of anxiety. It uses HTML5 (Hypertext Markup Text Language) JavaScript, and CSS for the website's layout (Cascading Style Sheets)

SYSTEM ANALYSIS

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

- The existing system contain sorting visualization only for the random inputs.
- There is no related information about each of the sorting algorithm.
- We can only visualize with individual sorting technique. i.e., no comparison can be done between more sorting techniques.

2.2 PROPOSED SYSTEM

- The proposed system involves the simulation of the different type of sorting algorithms codes.
- Rearranging the size and speed as per the user. User can input his/her own elements to visualize.
- Comparison between two sorting algorithms at the same time.

2.3 FEASIBILITY STUDY

This project had been considered as an important tool for the visualization of sorting algorithms easily. The main objective of our team working on this project is to help people to understand and visualize this difficult sorting algorithms in an easy manner

SYSTEM REQUIREMENTS & SPECIFICATIONS

3.SYSTEM REQUIREMENTS & SPECIFICATIONS

3.1 HARDWARE REQUIREMENTS

- System : Intel Core i5&above
- Hard Disk : 512 GB or above
- Input Device : Keyboard and Mouse
- Output Device : Monitor or PC

3.2 SOFTWARE REQUIREMENTS

- Operating System : Windows 7,10
- Web Technologies : HTML, CSS, Java Script
- IDE Used : VSCode

3.3 FUNCTIONAL REQUIREMENTS

A Functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behavior, and outputs. Functional requirements may be calculations, technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish

- Functional Requirements concerns with the specific functions delivered by the system.
- The functional requirements of the system should be both complete and consistent.
- Completeness means that all the services required by the user should be defined.
- Consistency means that requirements should not have any contradictory definitions.

The requirements are usually described in an abstract way. However, functional system requirements describe the system function in detail, its inputs and outputs, exceptions and so on.

SYSTEM DESIGN

4.SYSTEM DESIGN

4.1 INTRODUCTION

System Design can be best understood from design goals and system architecture. Each and every system has its own system goals and significance.

DESIGN GOALS:

Few system design goals are as follows:

1. **Performance requirement:** All data entered shall be up to mark and no flaws shall be there for the performance to be 100%.
2. **Platform constraints:** The main target is to generate an intelligent system to predict the disease.
3. **Accuracy and Precision**: Requirements are accuracy and precision of the data given as input as well as produced as output.
4. **Modifiability:** Requirements about the effort required to make changes in the software. (Person- months).
5. **Portability:** Since mobile phone is handy so it is portable and can be carried and used whenever required.
6. **Reliability**: Requirements about how often the software fails. The definition of a failure must be clear. Also, don't confuse reliability with availability which is quite a different kind of requirement. Be sure to specify the consequences of software failure, how to protect from failure, a strategy for error Prediction, and a strategy for correction.
7. **Security**: One or more requirements about protection of your system and its data.
8. **Usability:** Requirements about how difficult it will be to learn and operate the system. The requirements are often expressed in learning time or similar metrics.

4.1.1 THE USER INTERFACE

The User Interface The design and structure of the user interface components has remained unchanged even if the underlying back-end code was refactored midway through the construction. The canvas area is where the four sorting algorithms are visualized, and that area will be the location where the sorting algorithms' output is edited in. The first row of four blue-bordered buttons at the bottom of the canvas are the selectable algorithms: Selection Sort, Bubble Sort, Insertion Sort, Merge Sort and Insertion Sort.

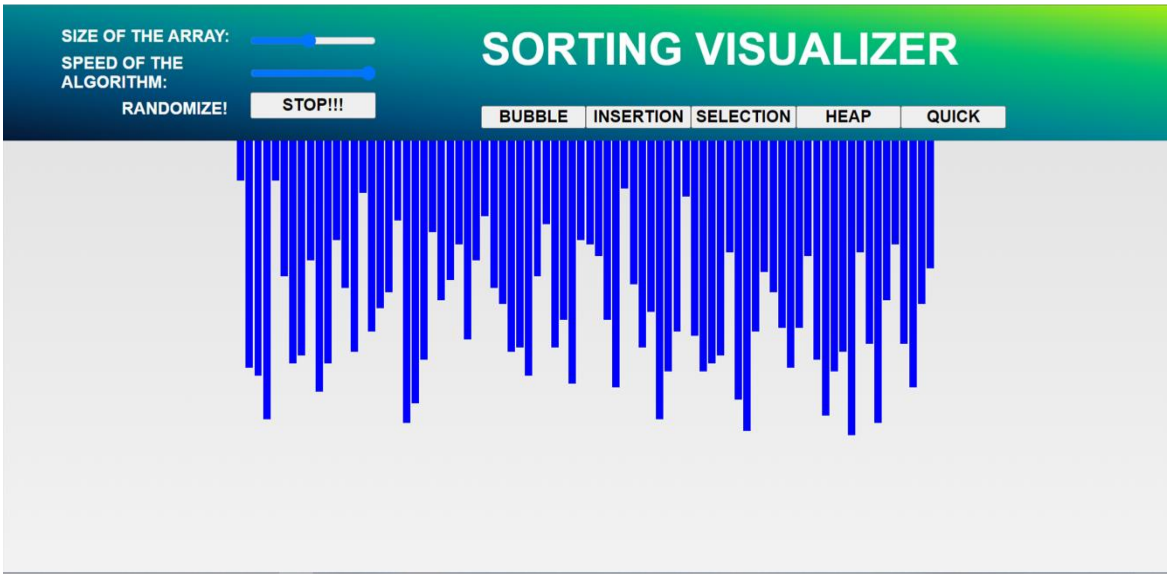


Figure 4.2 User Interface for Sorting Visualizer

The sorting algorithm is picked once the input and sorting method have been selected. Once the process is already underway, you can simply stop it by pressing the “Stop” button. The algorithm buttons are on separate tiers and have a blue coloration associated with them.

4.1.2 SYSTEM ARCHITECTURE:

HTML5, CSS, and JavaScript make up the back-end code. There are three varieties of code in one .html file and all three can be executed from this file alone. Including different types of web languages in a single page is one of the shortcomings of HTML.

Readability and keeping relevant code together are benefits of excellent programming practices. However, in the end, we opted not to break the code into two separate sections because of these two reasons.

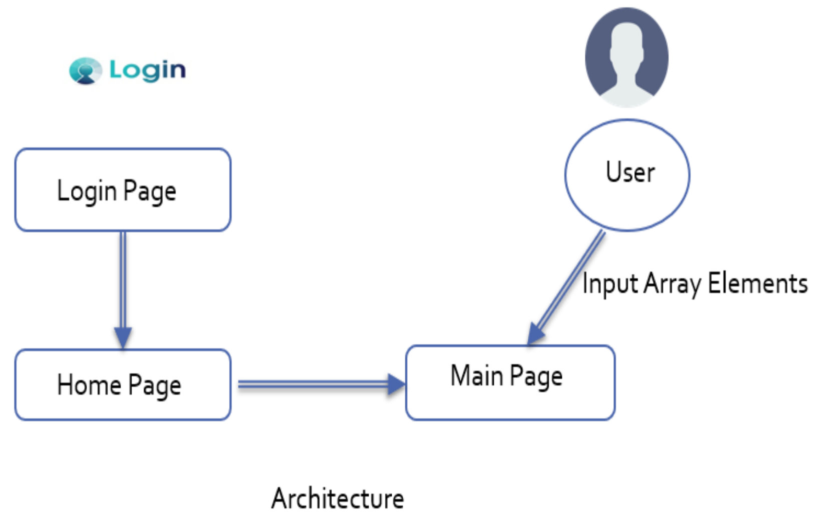


Figure 4.2 System Architecture for Sorting Visualizer

As you can see, the three coding languages are the only important components. However, since JavaScript runs immediately in the browser, it is unnecessary to employ a server on the back-end (like PHP). HTML5 and CSS are employed in web development. As illustrated with a single, bidirectional arrow, the HTML5 and JavaScript communicate to run the relevant algorithms and update the interface.

4.2 UML DIAGRAMS

Use Case Diagram:

It represents the functionality of a system by utilizing actors and use cases. It encapsulates the functional requirement of a system and its association with actors. It portrays the use case view of a system. Following are the purposes of a use case diagram given below:

- It gathers the system's needs.
- It depicts the external view of the system.
- It represents the interaction between the actors.
- It recognizes the internal as well as external factors that influence the system.

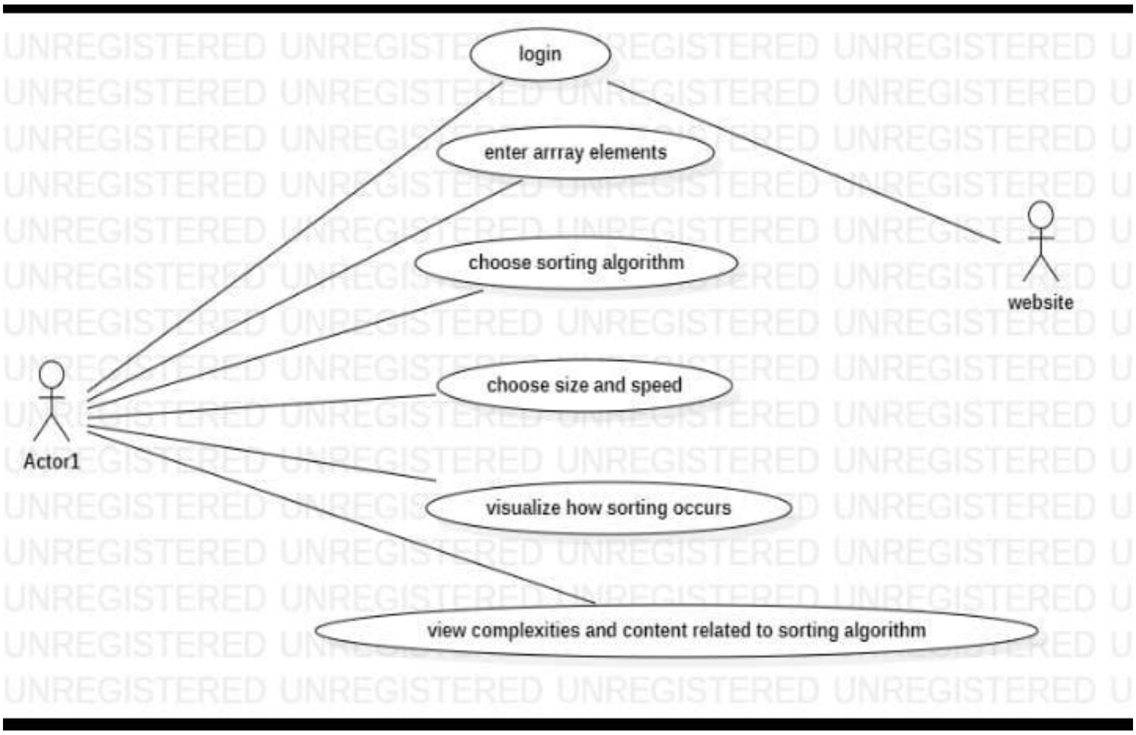

login
enter arrray elements
choose sorting algorithm
website
Actor1
choose size and speed
visualize how sorting occurs
view complexities and content related to sorting algorithm

Figure 4.2.1 Use Case diagram for Sorting Visualizer

Class Diagram:

Class diagrams are one of the most widely used diagrams. It is the backbone of all the object-oriented software systems. It depicts the static structure of the system. It displays the system's class, attributes, and methods. It is helpful in recognizing the relation between different objects as well as classes.

Following are the purposes of a class diagram given below:

- It analyses and designs a static view of an application.
- It describes the major responsibilities of a system.
- It is a base for component and deployment diagrams.
- It incorporates forward and reverse engineering.

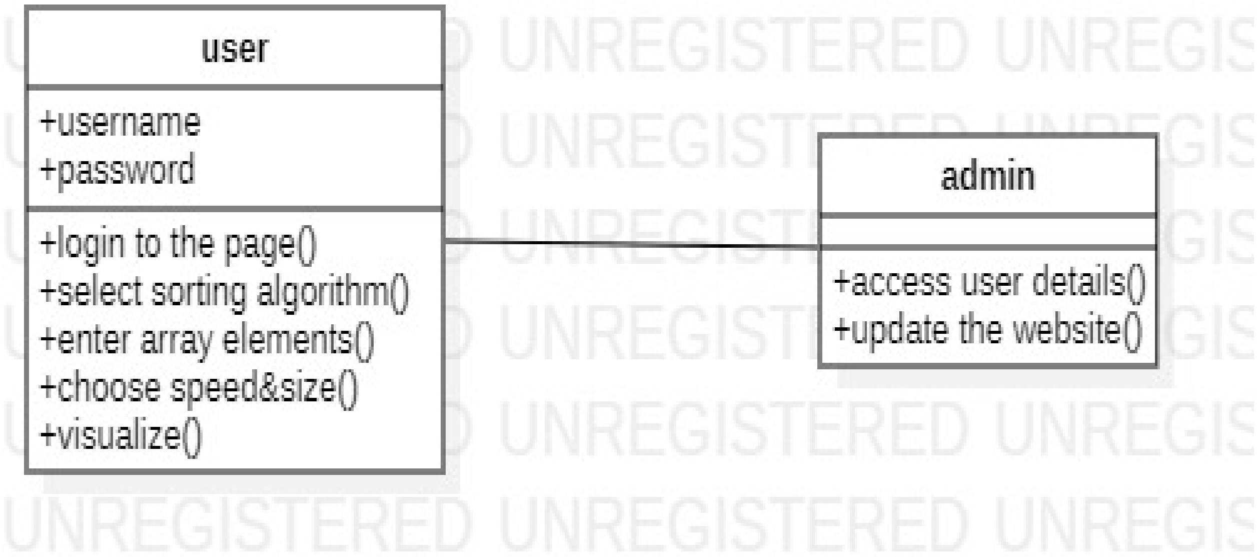


Figure 4.2.2 Class diagram for Sorting Visualizer

Sequence Diagram:

It shows the interactions between the objects in terms of messages exchanged over time. It delineates in what order and how the object functions are in a system.

Following are the purposes of a sequence diagram given below:

- To model high-level interaction among active objects within a system.
- To model interaction among objects inside a collaboration realizing a use case.
- It either model generic interactions or some certain instances of interaction.

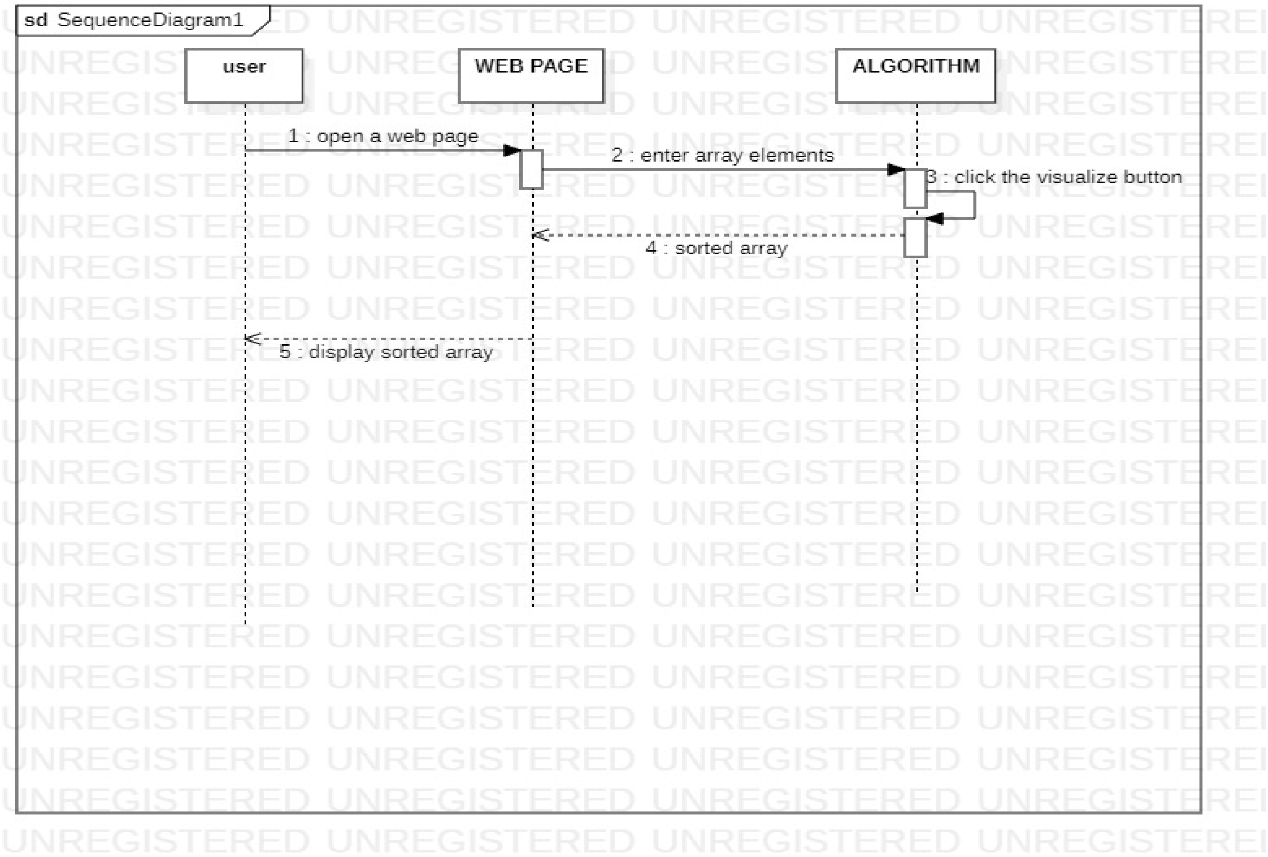


Figure 4.2.3 Sequence Diagram for Sorting Visualizer

Activity Diagram:

The basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system. The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

- Describe the sequence from one activity to another.
- Draw the activity flow of a system.
- Describe the parallel, branched and concurrent flow of the system.

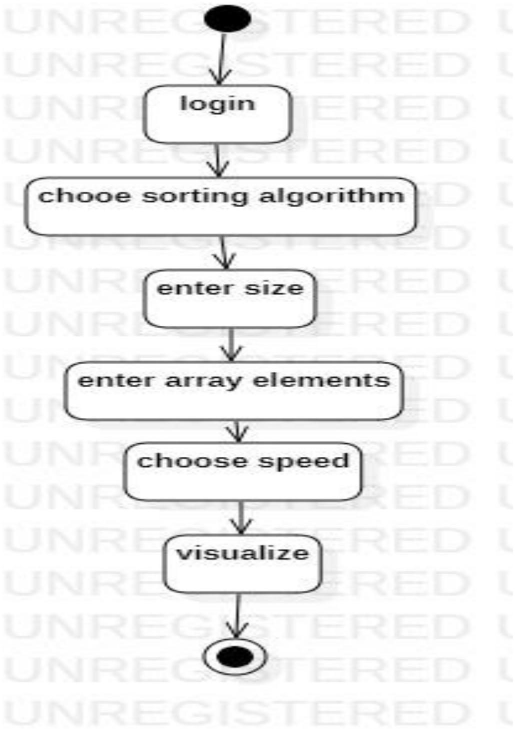


Figure 4.2.4 Activity Diagram for Sorting Visualizer

SYSTEM IMPLEMENTATION

5.SYSTEM IMPLEMENTATION

5.1 INTRODUCTION

The use of HTML, JavaScript, and CSS combine to form this project's implementation. There is only one project file which is an HTML file and contains the code. This software uses both object-oriented and functional programming paradigms in how it organizes the code. Before the final phase of development, the design was almost completely functional, where only three objects were used: one to control the canvas that displayed the animation, another to represent a piece of data, or "bar" object (blue rectangle with dynamically changing height and position).

This incorporated several function calls, some instance variables and Boolean values were utilized to keep track of the algorithm picked and when to animation, but this led to a greatly integrated mass of code that was difficult to maintain. Several big refactoring later, the code has now taken on the form of a Model-View-Controller Architecture.

Although, because of its functional character, it possesses a multitude of individualized functions that alter the instance variables and Boolean values, which means it has a multitude of functions that directly alter the View and Controller. The major module in the HTML code between the tags is known as the global scope. Everything within the framework is able to access the aforementioned variables and method.

5.1.1 The View:

There are three items on the view: the sort Area, the bar, and the position. These objects operate within the container defined by the tags in the .html file. This function's space is sometimes referred to as the "main" function, the first function invoked in a program. It is the sort Area that keeps the bars up to date using a timer, while at the same time generating the bar graph. As a result, whenever "Step" is invoked, the bar values are updated depending on the steps array. In the sort Area, after every second iteration of the timer, the rectangles will be redrawn with varied heights that represent the new values.

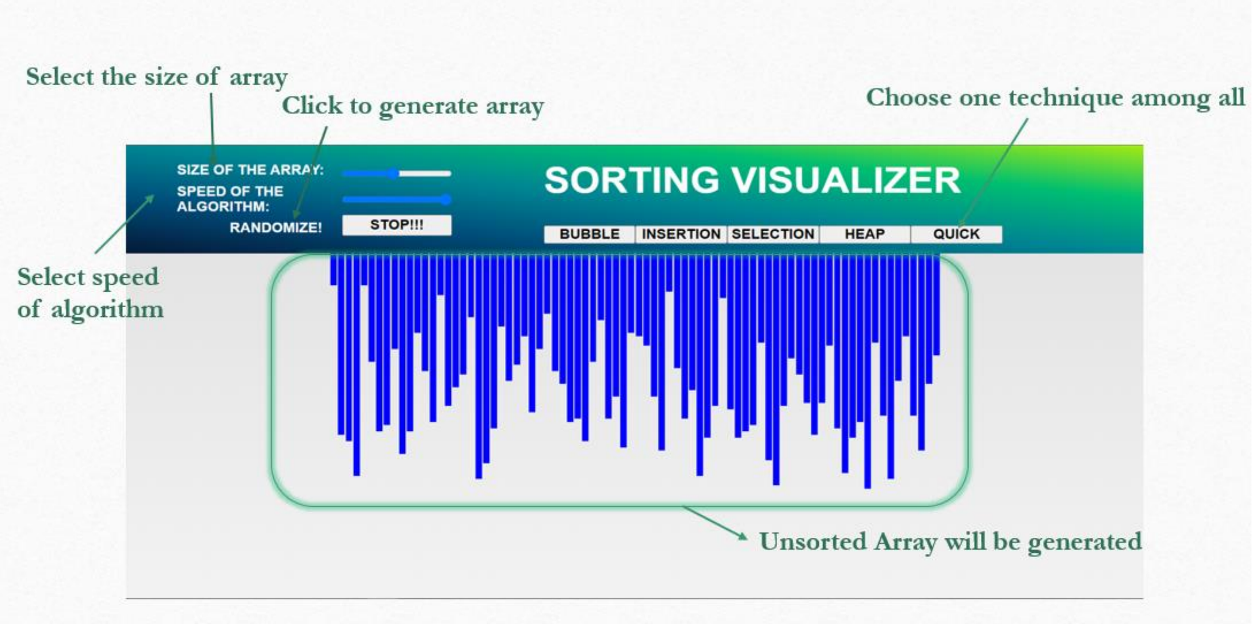


Figure 5.1 View of Sorting Visualizer

The bars change sixty times per second, so when the “Sort” button is selected, the change is instantaneous. In the sort Area, the bar object represents each piece of data. While having a distinct array named bars for the current bars in the bar graph helps preserve attributes such as the total number of bars (total value) independent of other characteristics, it is simple to update any or all of the attributes by iterating over the array as necessary.

5.2 PROJECT MODULES

Home Page:

This module consists of the home page where it the main page where the user can view initially and having some information like sorting and its techniques and all information about the sorting and it's overview. We implemented the home page with HTML, CSS and some BOOTSTRAP for the user-friendly experience with some animations and there are some buttons like sort, about which is redirected to its own page respectively.

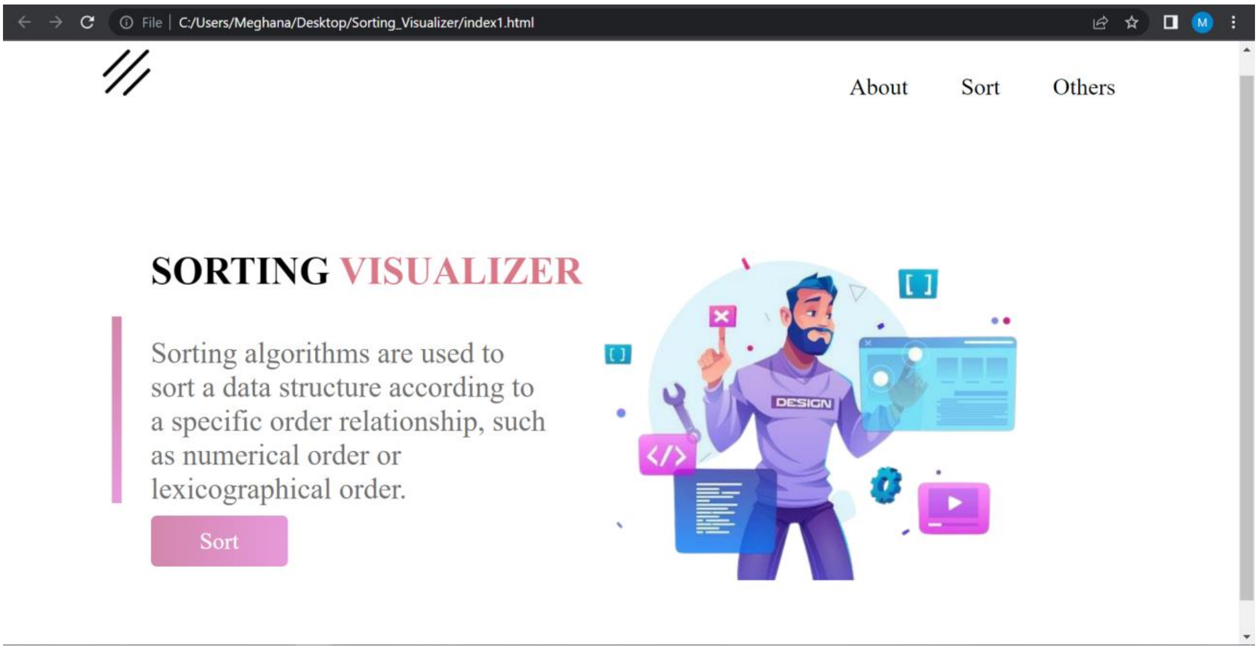
Main Page:

This module consists of the main heart of the project where the main sorting visualization happens here. This page is build using HTML, CSS and an inbuilt BOOTSTRAP and for visualization purpose JavaScript. In this module user can easily experience the visualization because of instructions provided.

About Page:

This module consists of the importance of each sorting algorithm clearly using its time and space complexities individually.

5.3 SCREENS



SORTING VISUALIZER

Sorting algorithms are used to sort a data structure according to a specific order relationship, such as numerical order or lexicographical order.

Figure 5.2 Home Page of Sorting Visualizer

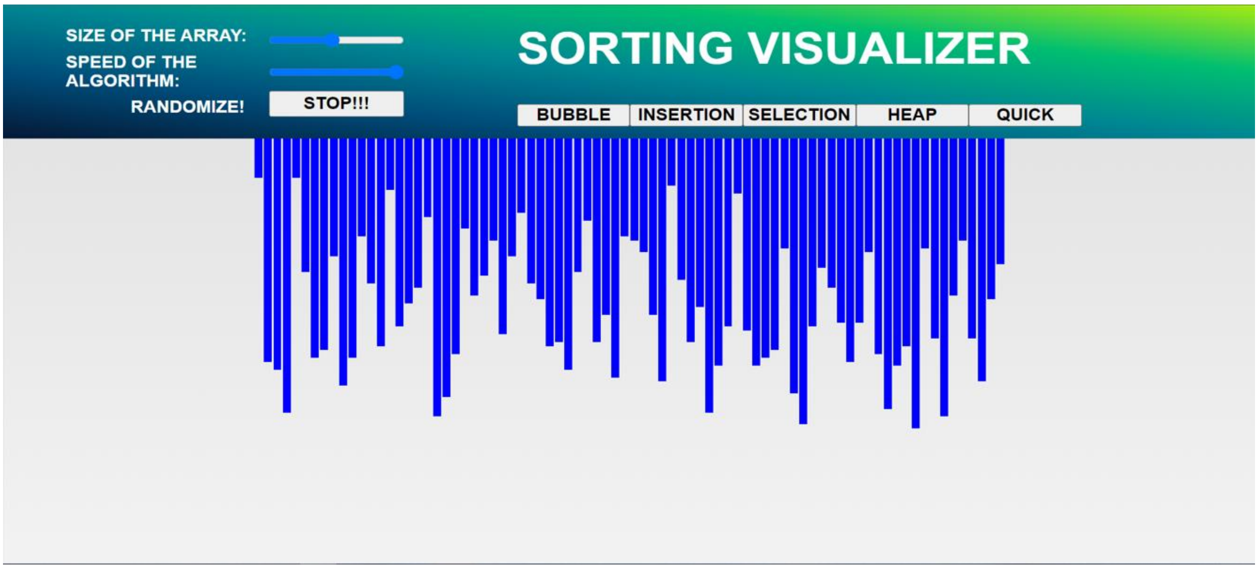
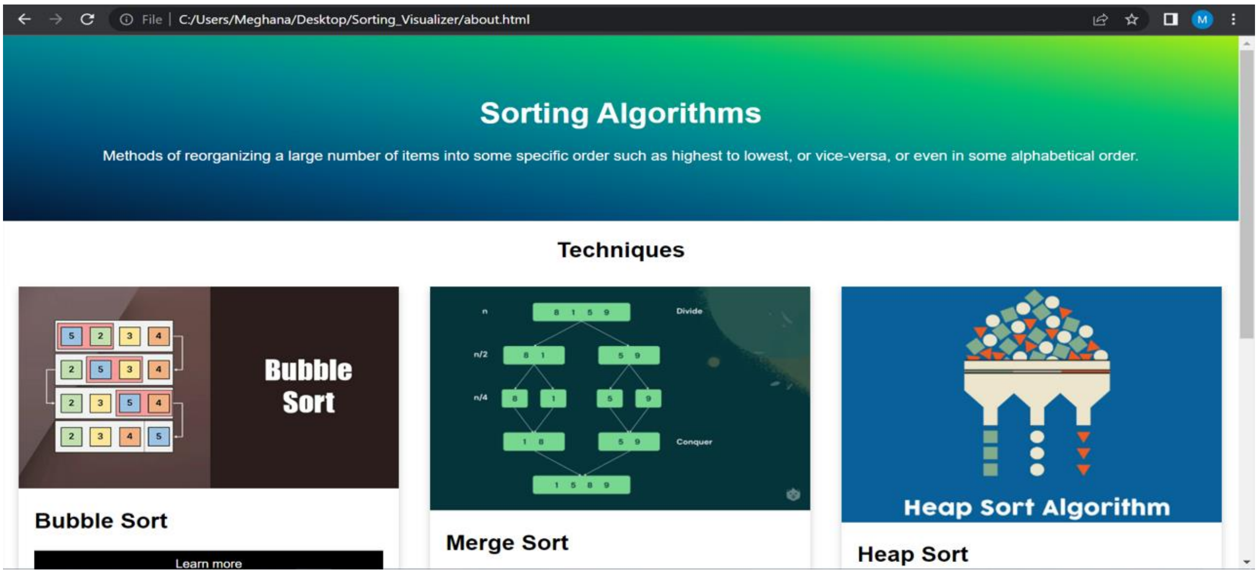


Figure 5.3 Main Page of Sorting Visualizer


File | C:/Users/Meghana/Desktop/Sorting_Visualizer/about.html
Bubble Sort
n
n/2
n/4
Divide
Conquer
Heap Sort Algorithm

Sorting Algorithms

Methods of reorganizing a large number of items into some specific order such as highest to lowest, or vice-versa, or even in some alphabetical order.

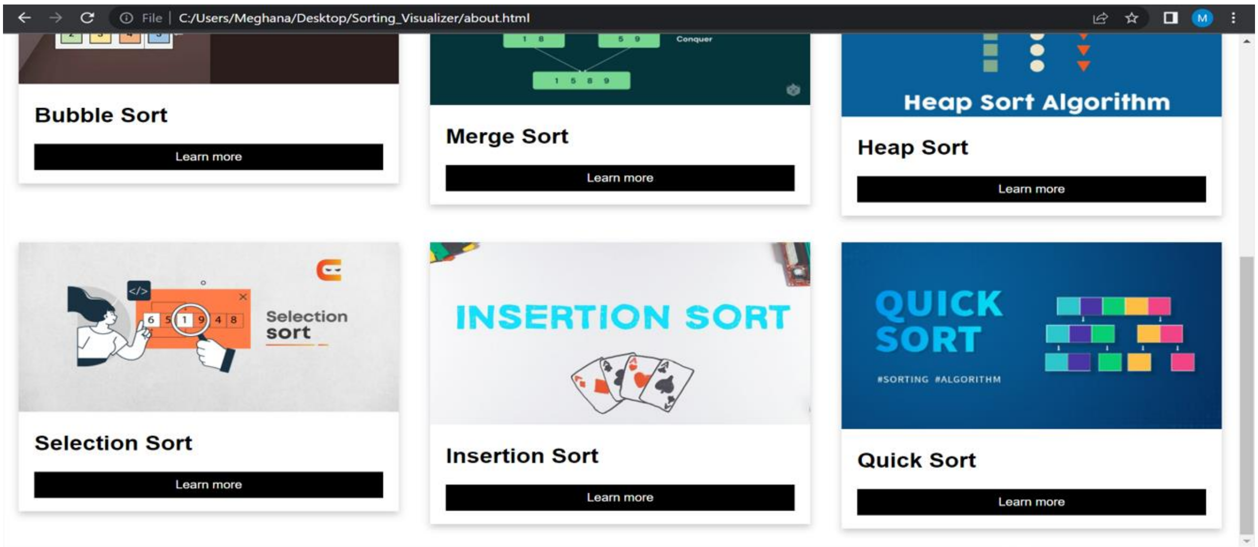
Techniques

Bubble Sort

Learn more

Merge Sort

Heap Sort



Bubble Sort

Learn more

Merge Sort

Learn more

Heap Sort

Learn more

Selection Sort

Learn more

Insertion Sort

Learn more

Quick Sort

Learn more

Figure 5.4 About Page of Sorting Visualizer

TESTING

6.TESTING

6.1 INTRODUCTION

In a software development project, errors can be introduced at any stage during development. Though errors are detected after each phase by techniques like inspections, some errors remain undetected. Ultimately, these remaining errors are reflected in the code. There are two types of approaches for identifying defects in the software—static and dynamic. The basic purpose of testing is to increase the confidence in the functioning of Software under test (SUT). Clearly, the effectiveness and efficiency of testing depends critically on the test cases selected.

Different levels of testing are used in the testing process, each level of testing aims to test different aspects of the system. The basic levels are unit testing, integration testing, and validation testing. These different levels of testing attempt to detect different types of faults.

6.2 TESTING METHODS

WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document.

UNIT TESTING

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

VALIDATION TESTING

The process of evaluating software during the development process or at the end of the development process to determine whether it satisfies specified business requirements.

Validation Testing ensures that the product actually meets the client's needs. It can also be defined as to demonstrate that the product fulfils its intended use when deployed on appropriate environment.

6.3 TEST CASES

HEAP SORT

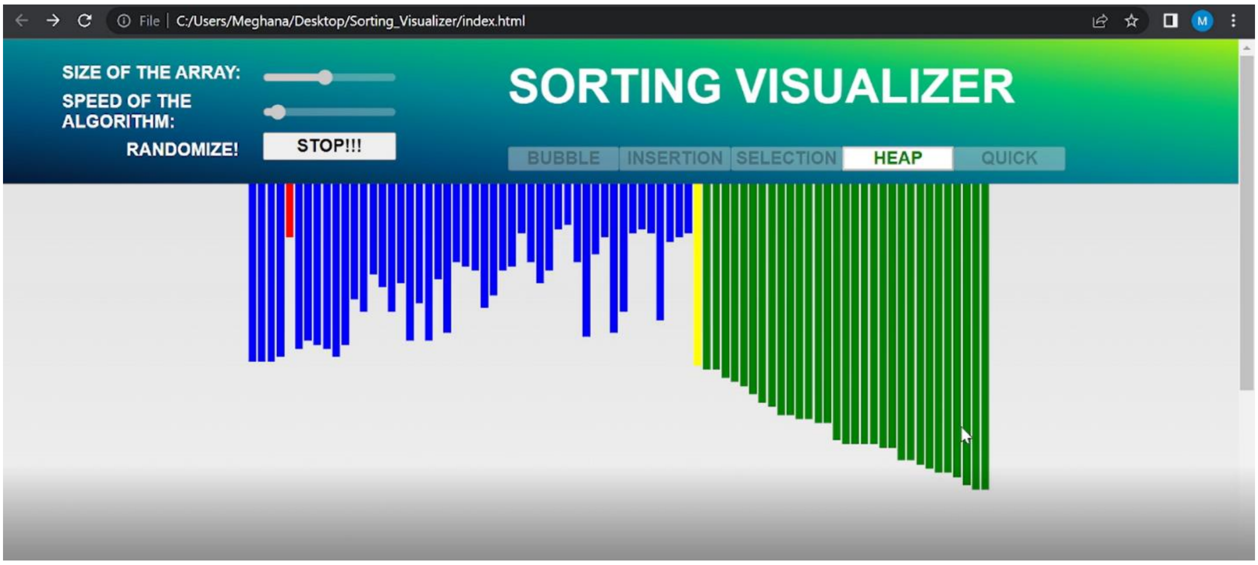


Fig:6.3.1 Test Case-1

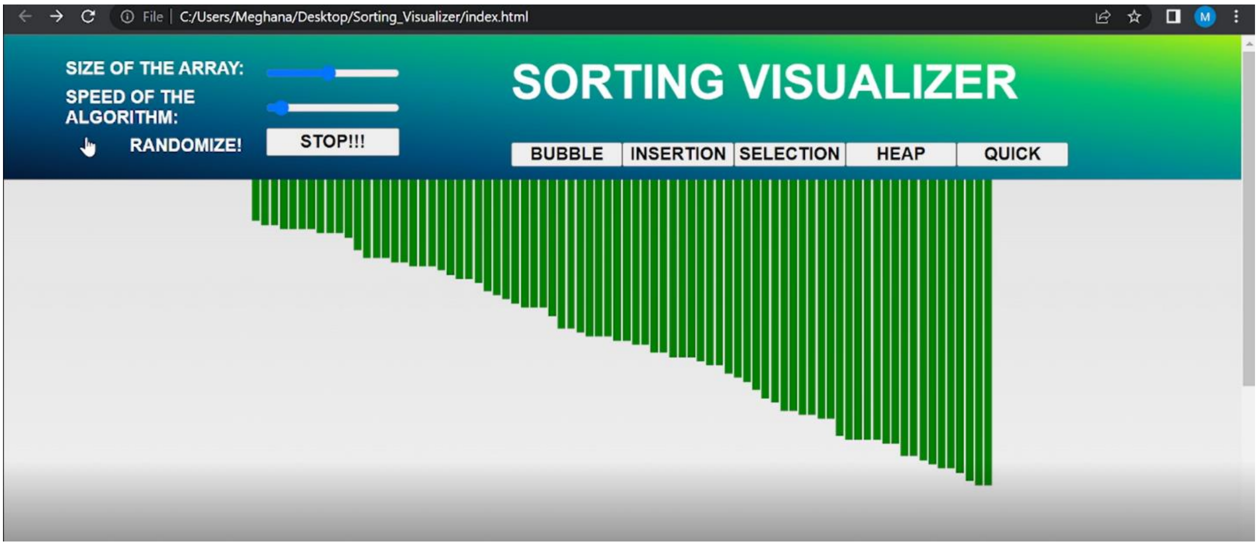


Figure 6.1 Heap Sort Testing

CONCLUSION

7. CONCLUSION

We have learnt sorting algorithms like bubble sort, insertion and merge sort, quick sort. But often we fail to understand the core idea of a particular algorithm Maybe because we are unable to visualize how they work.

So, the most important thing to understand about these algorithms is visualization. The sorting techniques are heart of the data structures where there are some sorting algorithms like quadratic and logarithmic algorithms which are bubble sort, insertion sort, selection sort, heap sort, quick sort and bucket sort etc., which have the complexities of O(n^2), O(n*logn), O(n), O(logn) and O(1). So, this visualization works really well for any sorting algorithm to understand easily.

BIBLIOGRAPHY

8.BIBLIOGRAPHY

1. https://visualgo.net/en/sorting

2. https://www.youtube.com/watch?v=JMnZ3VoWinY

3. https://www.geeksforgeeks.org/selection-sort-visualizer-in-javascript/

4. https://www.youtube.com/watch?v=cW16SGqr_Lg

5. https://clementmihailescu.github.io/Sorting-Visualizer/

APPENDIX

9.APPENDIX

9.1 Introduction to Web development

The three main languages you need to know well are HTML, CSS, and JavaScript. From there you can focus on frameworks, libraries, and other useful tools.

HTML

HTML stands for Hyper Text Markup Language. HTML displays the content on the page like buttons, links, headings, paragraphs, and lists.

Markup languages work in the same way as *you* just did when you labelled those content types, except they use code to do it -- specifically, they use HTML tags, also known as "elements." These tags have pretty intuitive names: Header tags, paragraph tags, image tags, and so on.

Every web page is made up of a bunch of these HTML tags denoting each type of content on the page. Each type of content on the page is "wrapped" in, i.e., surrounded by, HTML tags.

Once a tag has been opened, all of the content that follows is assumed to be part of that tag until you "close" the tag. When the paragraph ends, I'd put a closing paragraph tag: </p>. Notice that closing tags look exactly the same as opening tags, except there is a forward slash after the left angle bracket. Here's an example:

<p>This is a paragraph. </p>

Using HTML, you can add headings, format paragraphs, control line breaks, make lists, emphasize text, create special characters, insert images, create links, build tables, control some styling, and much more.

CSS

CSS stands for Cascading Style Sheets. CSS is responsible for the style of your web page including colours, layouts, and animations.

If HTML is the drywall, CSS is the paint.

Whereas HTML was the basic **structure of your website**, CSS is what gives your entire website its style. Those slick colours, interesting fonts, and background images? All thanks to CSS. This language affects the entire mood and tone of a web page, making it an incredibly powerful tool -- and an important skill for web developers to learn. It's also what allows websites to adapt to different screen sizes and device types.

JavaScript

JavaScript allows users to interact with the web page.

JavaScript (JS) is a lightweight, interpreted, or just-int-time compiled programming language with the first-class functions. While it is most well-known as the scripting language with for web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB, Adobe Acrobat.