LNMIIT

IMAD PROJECT REPORT

GROUP: 4

ALGORITHM VISUALISER

Team Members :

Aryan Aditya (20ucs031)

Payal Chhabra (20dcs006)

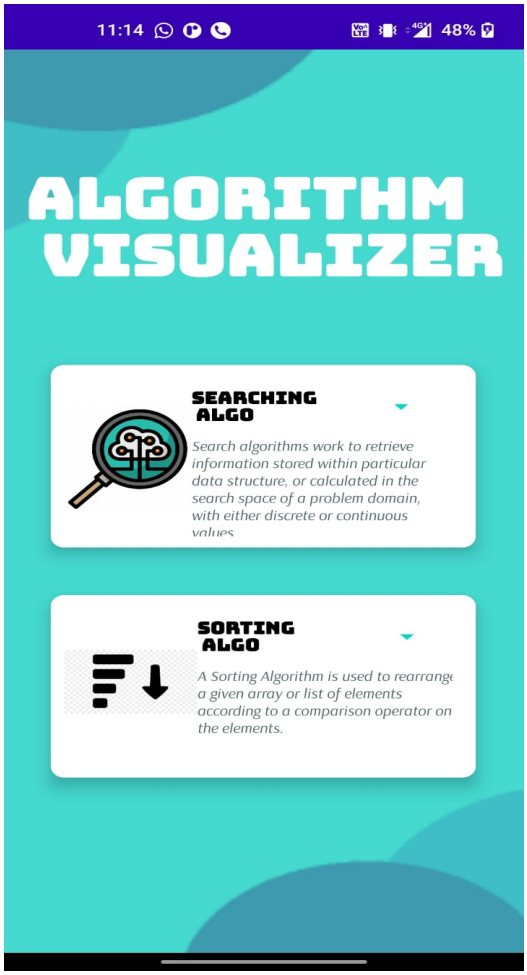
Chitransh Jaiswal (20ucs059)

Prateek Jain (20ucs145)

Janmejay Rathore (20ucs087)

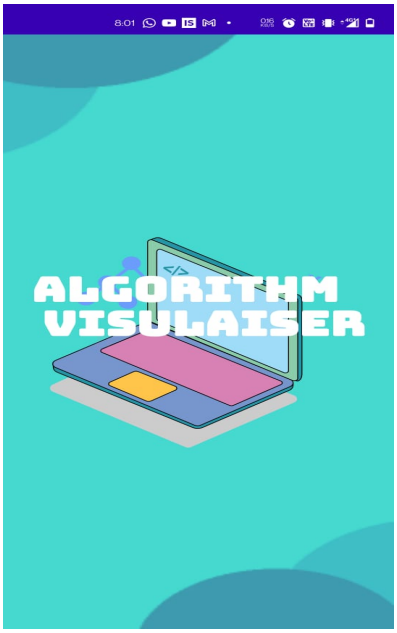
Product Overview

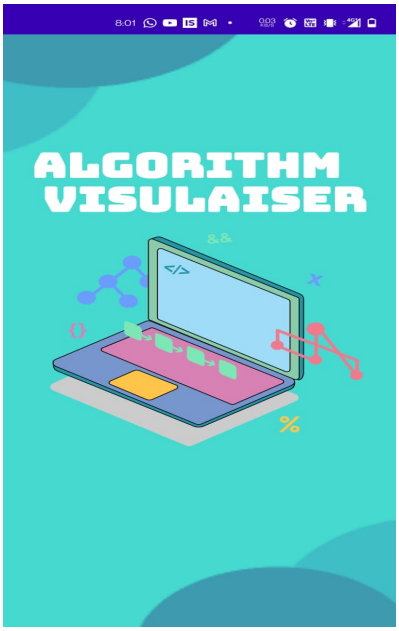
The product is software referred to as "Algorithm Visualiser". The application's goal is to animate data sets to improve users' ability to see data for better comprehension. The software presents a number of possibilities from well-organized algorithms in accordance with their categories, from which one may choose to go further. Data structures and algorithms are incredibly important, but they are also among the most fundamental criteria that one must meet in any language. Our application strives towards a better comprehension model, and thus far, it has made significant progress. To improve the application and assist an increasing number of individuals, more improvements and the incorporation of far broader elements and models will be made in due time.



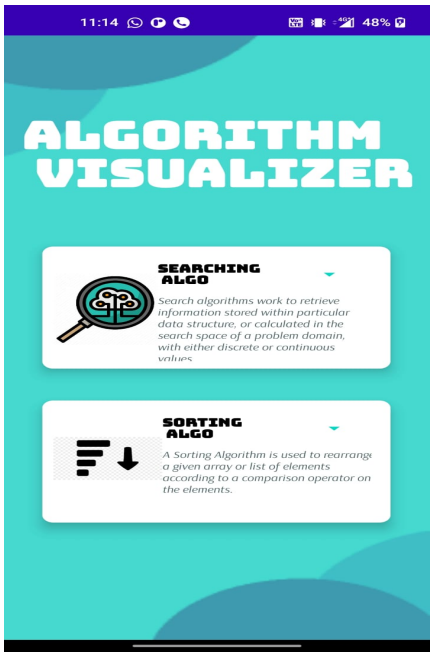
UI

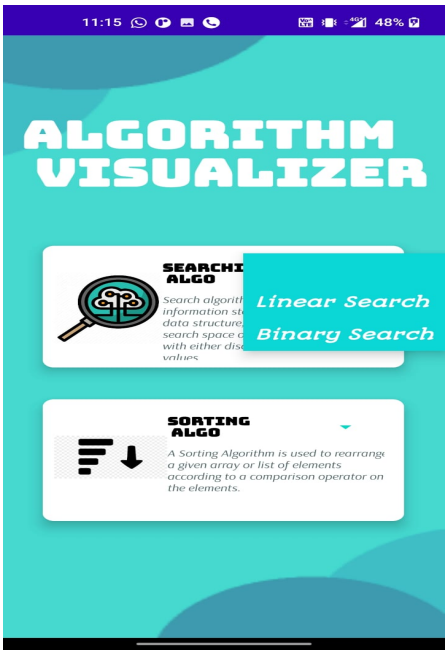
Animation:



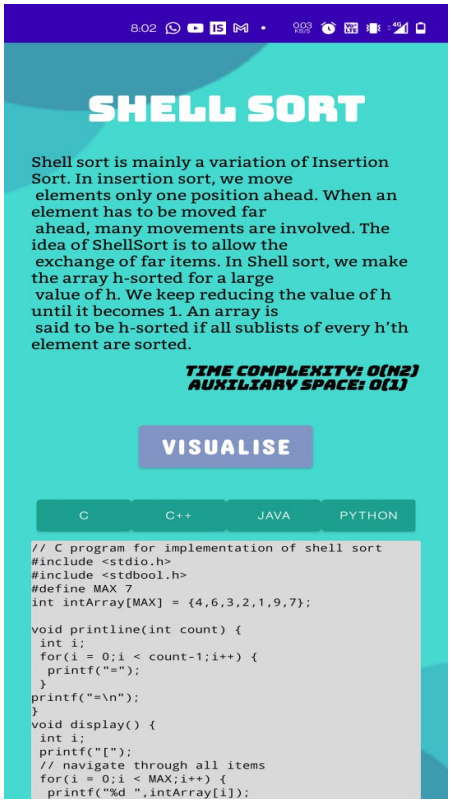


Home Page:

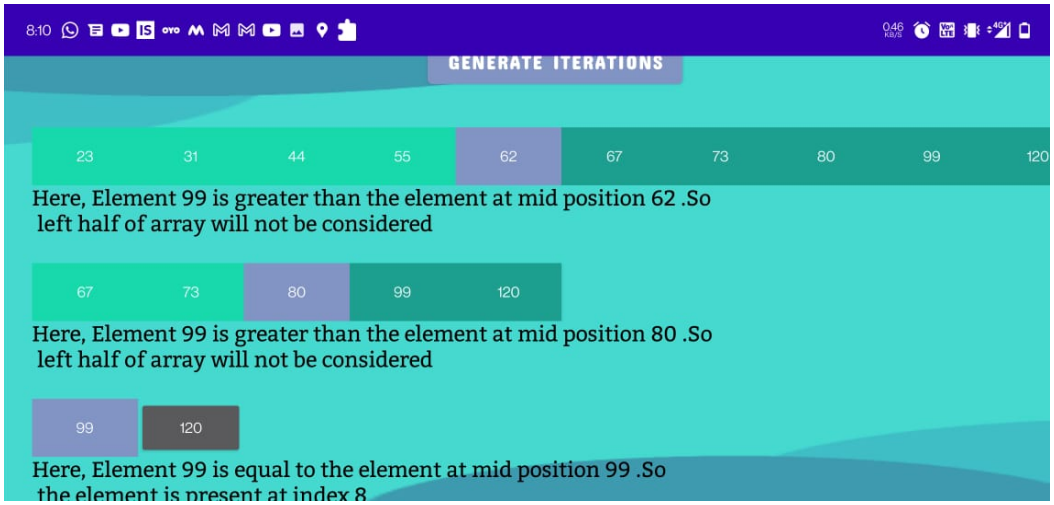


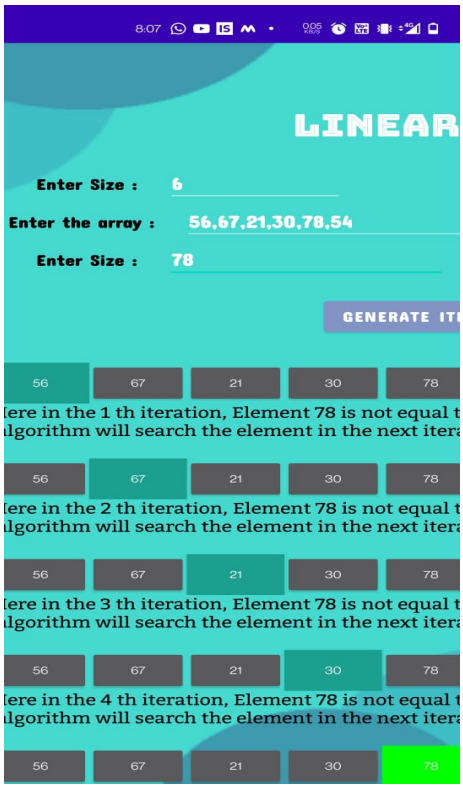


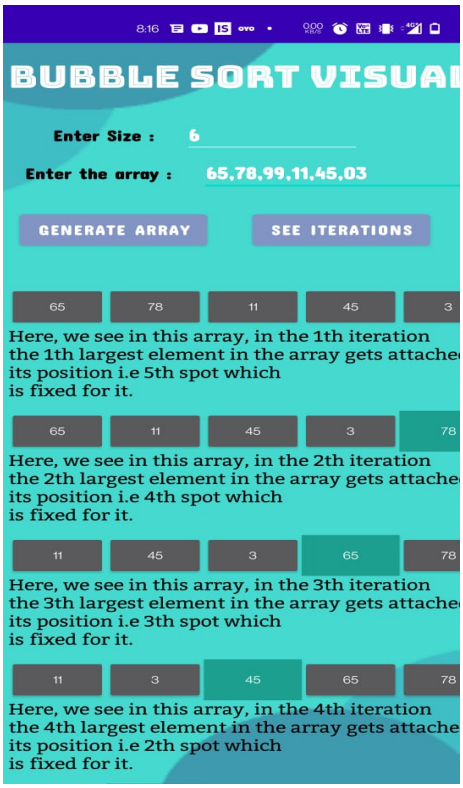
Algorithm details page :



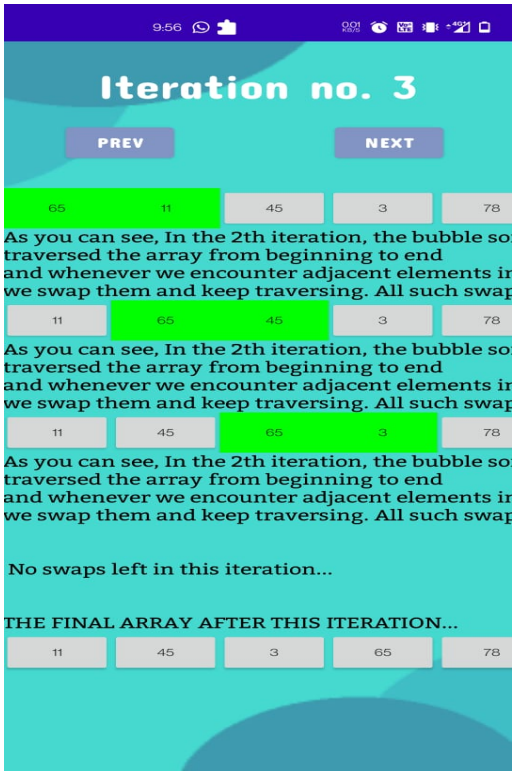
Visualize page:







Iterations page :



Interfaces

The program primarily comprises five user interfaces, three of which can change depending on the algorithm selected. The different interfaces have been briefly discussed here.

1. Splash Screen - A little animation that we identified to be related to our theme opens the application.
2. Home Page - From this point on, the project's true functionality begins. There are two categories, "Searching Algorithms" and "Sorting Algorithms," from which one can select. Both of these sections include a drop-down menu of their respective sorts, from which you may choose one to start a new page.
3. Description Page - A brief summary of the algorithm is presented on the description page after the user has chosen a specific algorithm. Its components are the following: Definition, Overview, Space and Time Complexity, and Code in C, C++, Java, and Python. To receive a complete and accurate view, users can scroll both horizontally and vertically when switching between the coding languages. The "Visualize" button the user is given takes them to more features.
4. Visualiser Page - It is the very heart and soul of this whole project where the most important code comes into place. The user has to enter the size of the array followed by the array’s data. Data entered can be separated by either commas or spaces. The program has been configured to read them as breakage signs. In the case of searching algorithms, target data also needs to be entered. Once the required data is entered, the user is provided with the “Generate array” and “See iterations” options. On selecting the former, the horizontal view beneath the buttons shows a step-by-step procedure with visual animations, linguistic explanations, and diagrams to show the algorithms' working process. On choosing the latter one, the user is directed to another page.
5. Iterations Page - Here, a heading with the current array is displayed on the page. There are two choices below it: "Prev" and "Next." The user can choose one of these to advance or rewind through iterations. Below them, a quick explanation is provided for the user to read, and the array's every iteration is expanded deeper into each swap operation to provide more clarity. To avoid confusion, the animations are colored, and the color code is also visible. The last array shown after the current iteration becomes the heading for the following iteration. If the user clicks "Next" once more after reaching the final

iteration, a pop-up message advising them of the same appears, and they are taken back to the "Visualiser Page."

Person Wise Division of Work

Aryan Aditya - 20UCS031

Overall App Integration, Management and preparing codes and layouts of Bubble Sort and Merge Sort

Chitransh Jaiswal - 20UCS059

Codes and functionalities of Radix Sort and Insertion Sort

Janmejay Singh Rathore - 20UCS087

Codes and layouts of Insertion Sort and Quick Sort

Prateek Jain - 20UCS145

Codes and layouts of Shell Sort and Selection Sort

Payal Chhabra - 20DCS006

Layouts and Iterations of Linear Search and Binary Search

Problems Faced

Several challenges occurred whilst completing this project. The team leader was the only one with prior experience in Application Development but that too briefly in Flutter which when compared to Java turned out to be a completely different framework. All the remaining members were absolute beginners in android development. Hence, the first and foremost challenge we faced was figuring out how to achieve the goal relating to display of algorithm functionings.

Our first approach towards this was to make the use of animations in android studio. We started working on this approach initially. We started going through the documentations and also several youtube channels for learning this. But going more and more deep into this made us realize that with the amount of animations and complexity required to achieve what we were seeking was really too tough for us at this stage. Infact, it was more of a graphics, designing and

animation thing than actual logical coding. Also, the material available at blog sites and youtube was more related to basic animations for UI only.

Then our second approach was to use the Paint and Canvas library and classes for android studio. The Canvas class holds the "draw" calls. To draw something, you need 4 basic components: A Bitmap to hold the pixels, a Canvas to host the draw calls (writing into the bitmap), a drawing primitive (e.g. Rect, Path, text, Bitmap), and a paint (to describe the colors and styles for the drawing). The Paint class holds the style and color information about how to draw geometries, text and bitmaps. But on detailed study of the same, we realized that there were scalability issues on working with these libraries. They didn't have much freedom to do a lot of unique things which were different. We needed this freedom as we were covering a lot of different animations.

The above two challenges really changed the path for us as it made us realize the limitations and complexities when working with so many different algorithms. So we made a major decision and decided to only focus on searching and sorting algorithms thereafter.

Now after searching for other solutions, we came upon the concept of Dynamic UI. This involved using buttons, TableRows and Table Views to store certain information of the given state and then display accordingly. This use of Dynamic UI was Coupled with Colour coordinations and step explanatory texts in text views to help display the algorithms both visually and logically. We kept working on this approach and made our first algorithm that was Bubble sort. This gave us confidence and an idea of how to complete our project. After this, we equally divided the rest of the algorithms and started working.

Another challenge we faced was that even though we had completed one algorithm, the remaining were not similar in any way. It wasn't the case that the same logic could be used in two different algorithms. Working with Dynamic UI had made us code bits in between the algorithm which involved different states and situations depending on the algorithm. FOr eg: In merge sort which was depth first, storing the right split and also displaying it simultaneously with the left split was tricky which required a lot of figuring out and a completely different approach was then taken for it which used coloured rows as steps and black and white division of space separated blocks to show division and merging phase. Like this one example, every algorithm has its own small uniqueness which was dealt with by the person in his own imaginative way.

Libraries used

- Lottie animations
- YoYo animations

Experience

Our project is related to data structure and algorithms. Algorithms and data structures are relevant to our research. One may choose from a list of a variety of algorithms and sorting

techniques to learn more about them. Albeit the final product of our project is incredibly intriguing, the learning process itself provided us with a plethora of scoring opportunities.

As a team, we first struggled to distinguish between this topic and a few others. The inspiration for our research, however, came from our mentor's advice and enthusiasm for this particular topic. Another salient point was the depth and breadth of expertise on this subject but the dearth of such software on the market. It is a universal truth that understanding data structures and algorithms is important for anyone working in our industry, yet few do so to the fullest extent. It was really disappointing to see a lack of good visualization and animation, everyone was given a task that was fairly equally distributed among them. Each team member contributed well and showed up on time. The team's enthusiasm and cooperation at each stage have been crucial to the project's success. To facilitate collaboration, each participant was tasked to provide three interfaces for each of their algorithms using a pre-established nomenclature. The three interfaces contained pages explaining iterations, input and display, and a brief discussion of the algorithms (including their temporal and spatial complexity). Finally, these were integrated in order to incorporate all of the algorithms into a single project.

The procedure resulted in the development of our fantastic project. We were happy with the results of everyone's efforts. The majority of our expectations were met by us. And we can state with absolute certainty that we learned an awful lot.

Project Code Link :

https://drive.google.com/drive/folders/1DtlMYvZKWYQN2UMxWRukAts_I774NpJz