MAJOR-1 PROJECT

SYNOPSIS
ON
ALGORITHM VISUALIZER

Submitted By:

Name	Roll No	Branch	SAP ID
Prajjawal Banati	R171217044	CSE-DevOps	500060722
Prakhar Aggarwal	R171217045	CSE-DevOps	500061466
Priyansh Gupta	R172217039	CSE-BigData	500062001
Sajal Sood	R171217051	CSE-DevOps	500062540

Under the guidance of
Mrs. Kalpana Rangra
Assistant Professor (SG)
Department of Cybernetics

SCHOOL OF COMPUTER SCIENCE
UNIVERSITY OF PETROLEUM & ENERGY STUDIES
Dehradun – 248007, 2019-2020.

Approved By

(Mrs. Kalpana Rangra) (Dr. Monit Kapoor)
Project Guide **Department Head**

School of Computer Science

University of Petroleum & Energy Studies, Dehradun

Synopsis Report (2020-2021)

Major I

PROJECT TITLE: Algorithm Visualizer

ABSTRACT

We present our findings on the state of the field of algorithm visualization, based on extensive search and analysis of links to hundreds of visualizations. We seek to answer questions such as how content is distributed among topics, who created algorithm visualizations and when, the overall quality of available visualizations, and how visualizations are disseminated. We have built a wiki that currently catalogs over 350 algorithm visualizations, contains the beginnings of an annotated bibliography on algorithm visualization literature, and provides information about researchers and projects. Unfortunately, we found that most existing algorithm visualizations are of low quality, and the content coverage is skewed heavily toward easier topics. There are no effective repositories or organized collections of algorithm visualizations currently available. Thus, the field appears in need of improvement in dissemination of materials, informing potential developers about what is needed, and propagating known best practices for creating new visualizations

INTRODUCTION

Data structure and algorithm visualizations and animations (hereafter referred to generically as algorithm visualizations) have a long history in computer science education. While the 1981 video “Sorting out Sorting” by Ronald Baeker was the first well-known visualization, ad hoc visualizations existed long before. The first recognized system for creating algorithm animations was BALSA [2] in 1984. Since then, hundreds of algorithm visualizations have been implemented and provided freely to educators, and scores (or hundreds) of papers have been written about them. It is widely perceived that algorithm visualizations can provide a powerful alternative to static written presentations (from textbooks) or verbal descriptions supported by illustrations (from lectures). There has been some debate in the literature as to whether algorithm visualizations are effective in practice. Some studies have shown the classic dismissal that is the downfall of most technological interventions in education: “no significant difference” [6, 9, 11]. Other studies have shown that algorithm visualizations can

indeed improve understanding of the fundamental data structures and algorithms that are part of a traditional computer science curriculum [13, 3, 7]. Certainly, many visualizations exist and are widely (and freely) available via the Internet. Unfortunately, the vast majority of those currently available serve no useful pedagogical purpose. So we see that (a) many algorithm visualizations exist, yet relatively few are of true value, and (b) algorithm visualizations can be demonstrated to have pedagogical value, yet it is also quite possible to use them in ways that have no pedagogical effect. These facts seem to imply that creating and deploying effective algorithm visualizations is difficult. There is a small body of literature that investigates how to create pedagogically useful algorithm visualizations (for example, [10, 15]). Yet, there is still much to be done before we are at the point where good quality visualizations on most topics of interest are widely available.

LITERATURE REVIEW

Algorithm animation systems have a long history in education. The first reference to algorithm animation was the famous video 'Sorting Visualizer Tutorial' which has been presented by Clement Mihailescu on YouTube. This 30 min video demonstrated the characteristics and operations of nine sorting algorithms, using animation and audio comments. Since then, many researchers have developed various visualization systems to depict the behavior of certain algorithms and assist students' deeper understanding of algorithmic and programming concepts [4, 10, 16, 20, 22]. Nowadays, visualization involves more than the visual presentation of information in various media (such as animations, text, static or dynamic pictures, diagrams, etc.). Most algorithm visualizations provide an interactive environment that elicits active student participation using informative animations and multiple representations. Additionally, an effective visualization should not merely attract the visual attention of the learner but should be designed to promote the cognitive attention and engagement of the learner. In general, there are two main categories of visualization that focus on education: program visualization and algorithm visualization.

OBJECTIVES

Since university classes I've always wondered what it would be like if I didn't have to learn from the example the teacher had on the slides. I think of edge cases and other scenarios and I want to see how an algorithm behaves in those instances. This has been so far only possible on paper. Due to a recent burst of algorithm-exposure (interview preparation) I went through the same things again, so I decided to try out what I can do. Our main objectives will be to show the code of the algorithm, show the raw data structure, show the abstract data structure, describe the algorithm steps as the teacher would, visualize the connections between all these

PROBLEM STATEMENT

When we start learning to code we may face a lot of issues in understanding the actual working of the algorithm and visualising it, due to which it may be very frustrating and we may lose the motivation to code, So to make students understand each iteration and make them familiar to the algorithm with the help of animations and could visualize each iteration. So this project aims to create a platform for students to visualize the algorithms. This project will be covering all types of sorting, searching and graph algorithms to make students visualize better about each algorithm.

METHODOLOGY

The model we have used in our project is Agile.

Sprint-1:- Deploying Sorting Visualizer Unit

Goals:-

- Implement Bubble Sort
- Implement Insertion Sort
- Implement Selection Sort
- Implement Merge Sort
- Implement Quick Sort

Sprint-2:- Deploying Searching Visualizer Unit

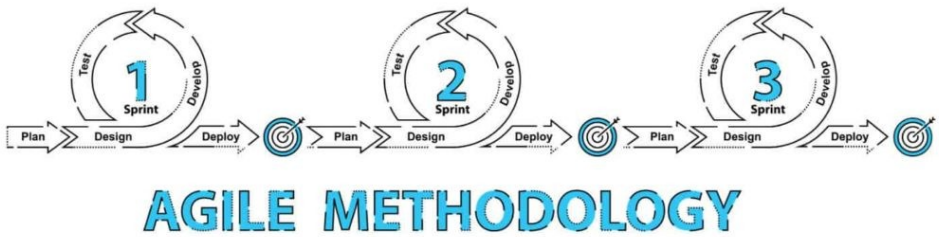
Goals:-

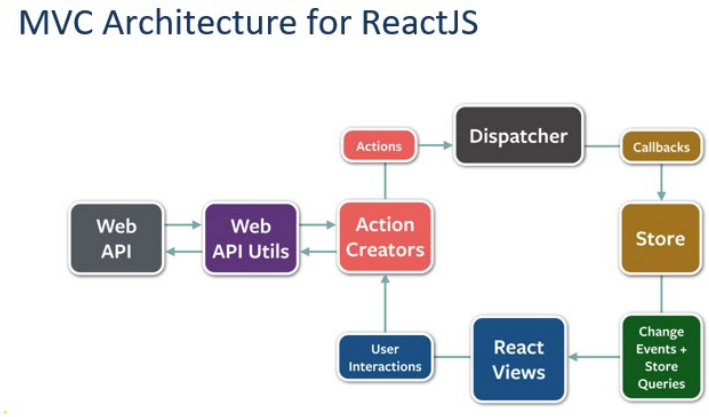
- Implement Linear Search
- Implement Binary Search

Sprint-3:- Deploying Graph Algorithm Visualizer Unit

Goals:-

- Implement Djakstra's Algorithm
- Implement BFS Traversal
- Implement DFS Traversal





SYSTEM REQUIREMENTS

Hardware Requirements:-

- 32/64 bit processor architecture supported by windows.
- Minimum RAM requirement for proper functioning is 8 GB.
- Minimum 64 GB free hard disk space depending on edition and configuration, including space required for files.

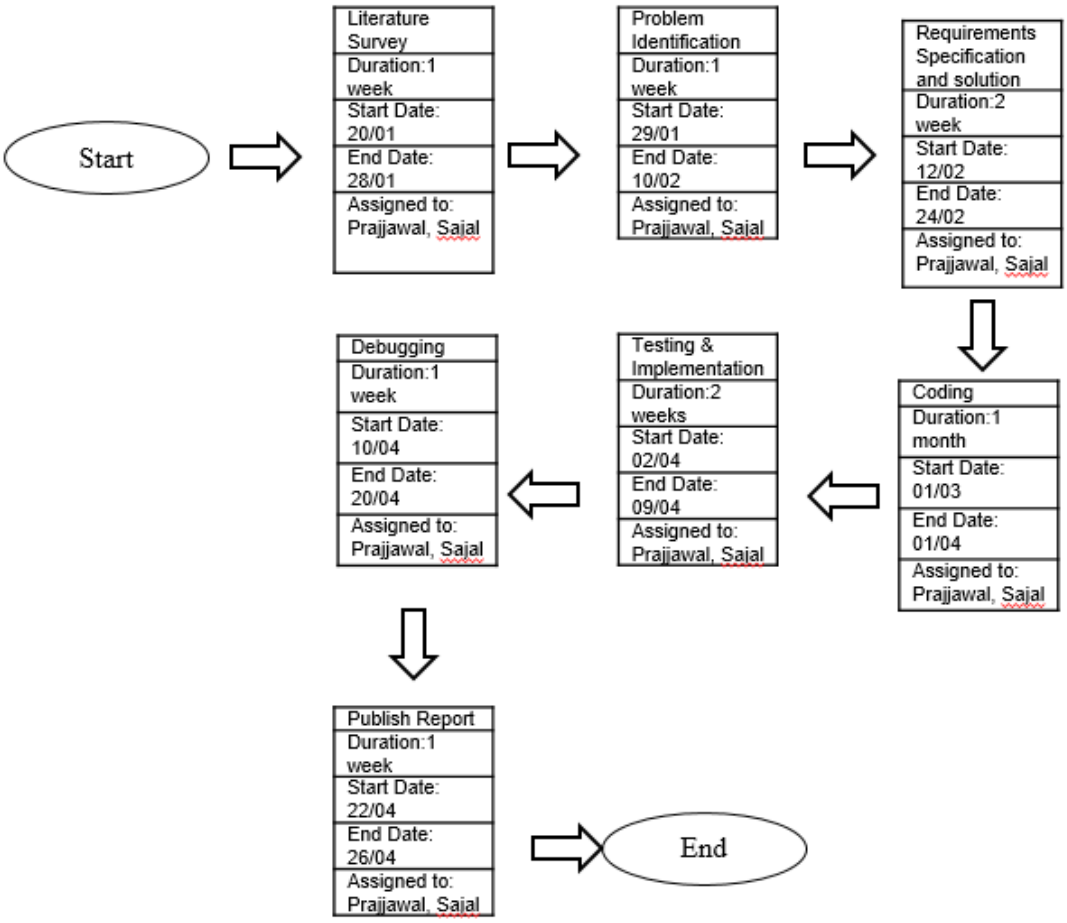
Operating Systems:-

- Windows (7 and above)
- Ubuntu (16.04 and above)

Software Requirements:-

- Npm(Node Package Manager)
- Create-React-App
- React Developer Tools(Chrome Extension).

SCHEDULE (PERT CHART)



REFERENCES

- Available at https://www.youtube.com/watch?v=pFXYym4Wbkc&t=89s [accessed on 14th Sep]
- Available at https://www.youtube.com/watch?v=msttfIHHkak [accessed on 14th Sep]
- Available at https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.134.3125&rep=rep1&type=pdf[accessed on 02nd Oct]
- Available at https://github.com/algorithm-visualizer/algorithm-visualizer [accessed on 04th Oct]
- Available at https://www.geeksforgeeks.org/sorting-algorithm-visualization-merge-sort/ [accessed on 04th Oct]