GG

Emmanuel Odame-Asante

1. **Introduction**

The first concepts of teaching have dated back to Ancient Greece. Vast knowledge from teachers get transferred to students. Students will grow and learn, thus expanding their own knowledge which is passed on to others; this cycle continues and fundamentally remains the same today. All this information, for thousands of years, has been transferred either orally or has been written down on a physical substance e.g. (stone, paper). However, as humanity advanced, there have been a plethora of ways different information can be transferred and assessed. The emergence of the *Information Age* is responsible for this.

With the *Information Age*, computers transformed the we live which effected every realm of our lives – teaching and information is part of this. We can now grasp unlimited information on any topic with a click of a button. This transition led to many new tools and applications that help us learn. This transition led to various tools and applications that are created to help understand the masses of information that is out there. Applications such as Khan Academy, Codecademy and Coursera are just a small subset of learning tool applications that exist.

Similarly to the concepts of teaching, the concept of algorithms has also existed for centuries. Greek mathematicians such as Euclid and Eratosthenes were studying the practises of algorithms. Algorithms have evolved significantly, especially with the dawn of the Digital Age there has been an expansion of the need of algorithms for our systems. Major applications such as Google Maps, Computer Networking are built on Graph Algorithms to exist. A category of algorithms that are used widely today are Graph Algorithms which are alone vital for our systems today. The overall aim of this project will be making a suitable Graph Algorithm learning tool.

* 1. Aims and Objectives

The core aim of this project is to create a tool which will efficiently aid users learning of significant graph algorithms. To create the most efficacious application as possible, intensive research of current technologies and practises will be undertaken. Time will also be invested on the User Experience design to make the application appealable to wide amount of users.

At the start of the project, I was assigned an Initial Project brief by Dr Sanderson. It as follows:

*“The aim of this project is to implement a tool to assist with the teaching of graph algorithms such as those taught in CE204.*

*It should provide a graphical display of step-by-step implementation of the algorithms with accompanying descriptive text. This should be offered at various levels of abstraction (e.g. for Kruskal's algorithm a version showing the connection sets and a more abstract version that uses the concept of not forming a cycle). It should be able to work on arbitrary graphs (i.e. the descriptions should not be based on a particular graph).*

*Ideally there should also be a graphical user interface that allows the user to create graphs by adding nodes at points on the display and creating edges to connect them, allowing the user's chosen graph to be used to demonstrate the algorithm.*

*The tool could be written as a web application or a stand-alone Java program.”* [1]

Additional Notes from Dr Sanderson:

*Decide what language and development environment you wish to use and whether you want to write a web application or a stand-alone program.*

*Try to develop a simple tool for generating displays of graphs.*

*Consider what algorithm(s) you’d like your tool to teach and research them.* [2]

Following the initial brief, key objectives were created that have been designated as fundamental to the project’s success are:

* Creating a robust, aesthetic web application that can effectively create nodes and edges on mouse click.
* ­Handles support for graph algorithms, including, but not limited to, Dijkstra’s, Kruskal’s and Prim’s.
* Informative text that will accompany the step – by – step execution of the graph algorithms.
* Emphasises on User Experience making sure that the application is as user friendly as possible.
  1. Scope

The scope of this project will be creating an effective application that will aid the teaching of algorithms that are taught in Dr Sanderson Data Structures and Algorithms course. Given time, the scope of the project can be extended to accommodate other algorithms such as A\*, BFS and DFS.

* 1. Project Motivation

I strongly desire helping others understand topics that may be confusing the grasp. So, this project naturally appealed to me. Graph algorithms may seem complex to understand but I firmly belief that with the right tools they are actually relatively easy to understand. This project will aim to create a tool which will make it easier for users to understand the Graph algorithms and make them less daunting to learn. The application would be successful if users access this application and it aids their understanding for Graph Algorithms. Upon creating this application, I will also be improving my technological skills as I will be furthering my knowledge in algorithms and also the skills necessary to build a web application. I will also be using and improving Soft skills such as communication, time management and planning to achieve a finalised product.

* 1. Audience

From the proposal Dr Sanderson assigned, the main audience for this tool would be second year students studying Data Structures and Algorithms in University of Essex. However, the would be open to anyone that has the interest of learning Graph Algorithms. This application will be accessible enough so that even users with limited to no Computer Science knowledge can navigate and use the application for education purposes.

* 1. Approach

There are various approaches taken when undergoing this project. In terms of design, one of the approaches taken is to look at other websites with good User Experience principles and incorporate them into this application. I will be making minimum viable products for every sprint with it being pushed to GitHub. I also wanted to be very transparent with the audience this application, so often I will show my application to Computer Science students and ask for feedback.

* 1. Report Structure
* Chapter 2 gives a background of all the main information revolved around this project.
* Chapter 3 Design and Implementation
* Chapter 4 Results and Evaluation
* Chapter 5 Project Planning
* Chapter 6 Conclusion and Further Ideas.

1. **Background**

*This chapter introduces the reader to the general concepts of Graph algorithms, the psychology of learning tools and UX design. It also evaluates similar Graph Algorithm teacher tool applications that exist.*

* 1. Graph Algorithms

Graph algorithms are simply ways to solve problems that are represented on a graph structure. You may ask, what is a graph? A graph is a collection of *nodes*. In a graph, there is a possibility of a *nodes* connecting to each other. They are connected via *edges*. (Fig 1.0) may have a property *weight.* Edge weight is a concept that illustrates the difficulty of traversing through that edge. The higher the weight, the ‘harder’ it is to traverse through the using that edge.

From these concepts, many questions can arise from graph. For example, is Node X reachable from Node Y? From Node P to Node Y, what is the shortest path distance between the two?

Graph algorithms are responsible for answering these questions. The answers to these questions are vital for our day to day systems. They have profound effect on the technology that we use today. For example, Dijkstra’s algorithm is one of fundamental algorithms that are used for directional problems on a virtual map such as application such as Google Maps. (Fig 1.1) Each points of interest may be represented as nodes and the streets or roads as edges. Asking Google Maps to give the shortest path between two points of interest will simply run a version of Dijkstra’s algorithm to calculate the shortest path.

Without Graph Algorithms, many of the technical problems that we encounter today will be difficult to accomplish.

5

2

**Example of a Directed Graph (Fig 1.0)**



**Google Maps: A popular application that uses Graph Algorithms (Fig 1.1)**

* + 1. Dijkstra’s Algorithm

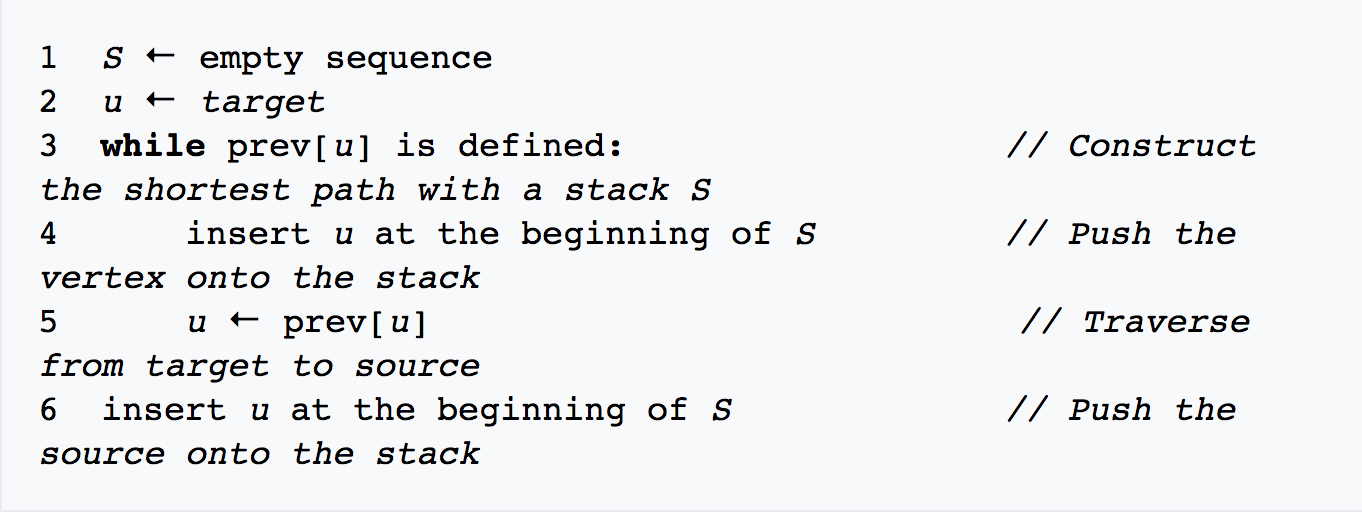
Dijkstra’s Algorithm is used as a solution to compute the shortest path between nodes on a graph. This was devised by Edgar Dijkstra in 1956. From the conception, however, it has been improved on many times to the algorithm that we have today, and it recognised as one of the most important algorithms in Computer Science. This algorithm is ubiquitous in the programming world and forms the foundation of optimization routing problems. [3]

Common real-world applications that Dijkstra’s algorithm are involved in:

* Geographic Information Systems which needs to determine the shortest path between point A to point B on a map. Example Google Maps.
* In Networking, the best way to move packets to a node.
* Telephone Network.

As mentioned before, it is highly influential and continues to dominate the way to solve the minimum cost path finding problems. This is defined as a Greedy algorithm, the formal definition of this being it’s a heuristic algorithm that at every step selects the best choice available at that step without regard to future consequences. [4] Dijkstra’s Algorithm worst case running time is: O(V log V).

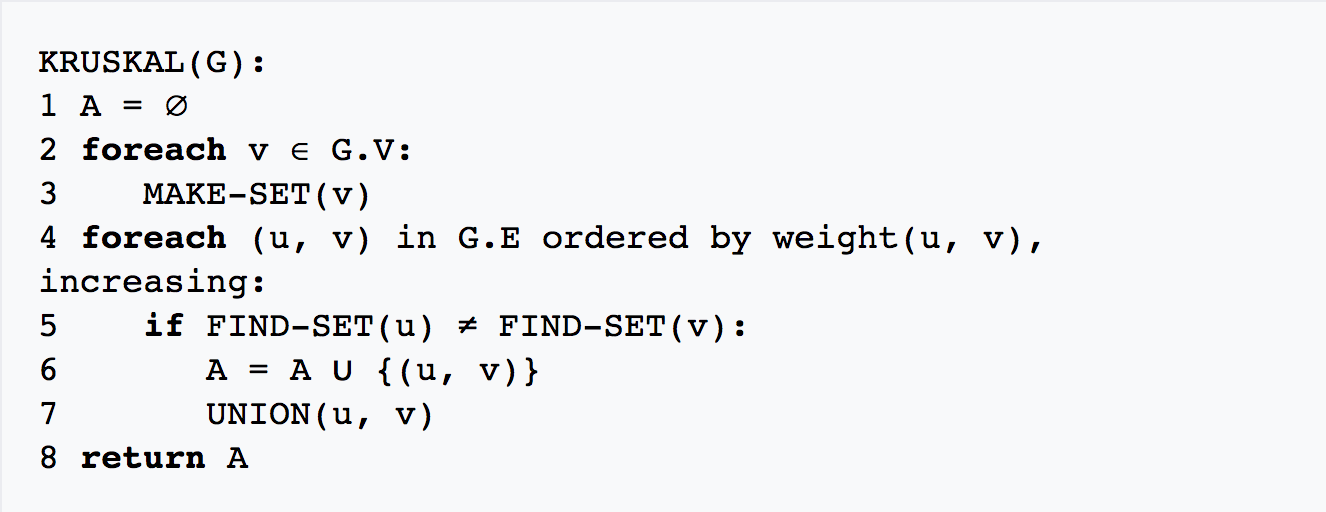
One of the disadvantages of Dijkstra’s algorithm is that It wastes significant time processing as it executes a blind search on the graph when it does not need to. So there are more efficient algorithms that exist that will not do this such as A\* algorithm. However, A\* does have its disadvantages too. Another issue is that Dijkstra also does not support negative edges.



**Pseudo-code implementation of Dijkstra’s Algorithm (Fig 2.0)**

* + 1. Kruskal’s Algorithm

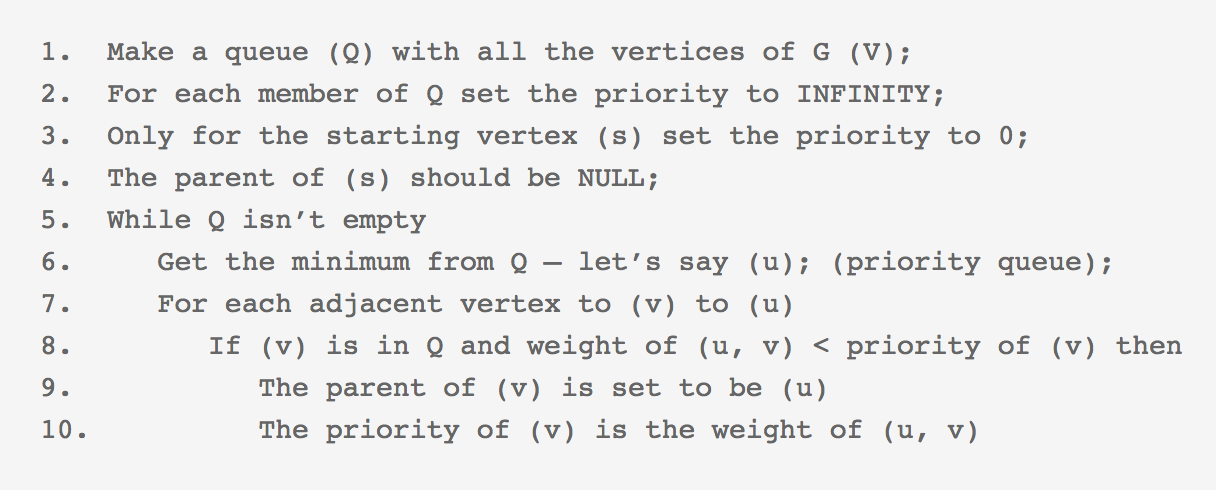
Kruskal’s algorithm is a greedy minimum spanning tree algorithm which finds an edge of the least possible weight that connects any two trees in the forest. [5] This algorithm was first exposed to the public in the Proceedings of the American Mathematic Society written by Joseph Kruskal in 1956. What it does differently to Prim’s that I will sort out all edges by weight and start adding to the MST starting from the smallest edge. The time complexity is . As a minimum spanning tree algorithm, the real-world applications for this are trying to build a telephone network. Kruskal’s algorithm is inherently sequential and hard to parallelize however it is possible to perform the in initial sorting of the edges in parallel.



**Pseudo-code implementation of Kruskal’s Algorithm (Fig 3.0)**

* + 1. Prim’s Algorithm

Prim’s algorithm is also a greedy minimum spanning tree that finds a minimum spanning tree for a weighted undirected graph. This was conceived by the Czech mathematician Vojtech Jarnik. How this differs from Kruskal’s algorithm is that there is a starting node which is accepted that will start the execution. Worst case time complexity is . As this is a minimum spanning tree algorithm, the same real-world applications are similar to Kruskal’s algorithm such as telephone network. In terms of parallel algorithmic calculations, the main loop of Prim’s algorithm is not parallelizable. However, the inner loop, which determines the next edge of minimum weight that does not form a cycle can be parallelized by dividing the vertices and edges between the available processors. [6]



**Pseudo-code implementation of Prim’s Algorithm (Fig 3.0)**

* 1. Learning Tools

As the genesis of this project is a teaching tool. It was wise to do background reading on what makes good teaching tool resources. The main points of what makes a good teaching tool are as follows:

* A good teaching tool is accurate

It is important for this graph algorithm tool to be accurate. It is especially vital the algorithms that are implemented are completely correct and the behaviour that is shown in the application is the same as the applications that are standardized in student’s lectures. The algorithms will have to be heavily tested to ensure that the algorithms implemented correctly, and they display the correct output. Without appropriate accuracy the application will be essentially useless, and users will seek elsewhere for aid.

* A good teaching tool is useful

The user should leave the application feeling satisfied with the time spent using the application. If the graph teaching tool is not useful for the user, then there will be no point of using them using it again in the future. The goal is to make sure that if the user is unsure about the algorithm when coming into the web application, upon them using it there would be an improvement of understanding of the algorithms they are inquiring about. A success would be that the application that is used by the user will be shared by the user to spread awareness of the application.

* A good teaching tool is efficient

I aim to make this teaching as efficient as possible. This will mean creating an application so users spend as little time as possible on trying to understand not only how the algorithms through the application but also spend as little time trying to understand how to operate the web application. This will come from high quality User Experience so that this efficiency will happen.

* 1. User Experience

User Experience is one the fundamental reasons for a successful customer facing application. User Experience refers to a person’s emotions and attitudes about using a particular product, system or service. It includes the practical, experiential affective, meaningful and valuable aspects of human-computer interaction and product ownership. Additionally, it includes a person’s perceptions of system aspects such as utility, ease of use and efficiency. [6]

Below highlights the principles of what makes a good User Experience in an application:

* Simple

The application should be simple to use. This means with little to no help, users should be able to successfully navigate around the web application. One goal of simplicity is that even without textual instructions, the user can navigate through the website and operate the functionality. This is the aim that the application will strive to get. There are common UX practises to what makes an application simple such as Intuitiveness. When operating the application, there should not be any actions that seem ‘strange’ to the standard user experience on the web. The operations executed by the website should be intuitive and familiar to the user so they understand what is going on.

* Aesthetic Appeal

Good User Experience, in my opinion, according to my research, one of the conditions that accompany good user design is attractive aesthetics. There may be a fully-fledged functional website however if it is not aesthetic at all, there may be a set of users that will not use the application. I would have to think about how the website looks so it’s attractive to my target audience. One of the most reliable ways to get in touch with the target audience would just be asking them. Researching on popular colour schemes on websites will be a good start to making an aesthetic application There could also be a survey that for example what will be the best colour schemes. I should be aware to not over-aesthetic the application as that can be detrimental the usability of the website. With research, it is shown that high colours with greater contrast ratio generally lead to greater readability. It was also shown that ratings of aesthetic quality significantly related to intention to purchase. [8]

* Usable

The application should be expected to be completely usable on most current platforms. The behaviour of the application aims to be consisted on all major browsers and also mobile phone. This will revolve intensive testing across many platforms. The Web application should accommodate users with disabilities so they can use the application too without major issues.

* Findable and Available

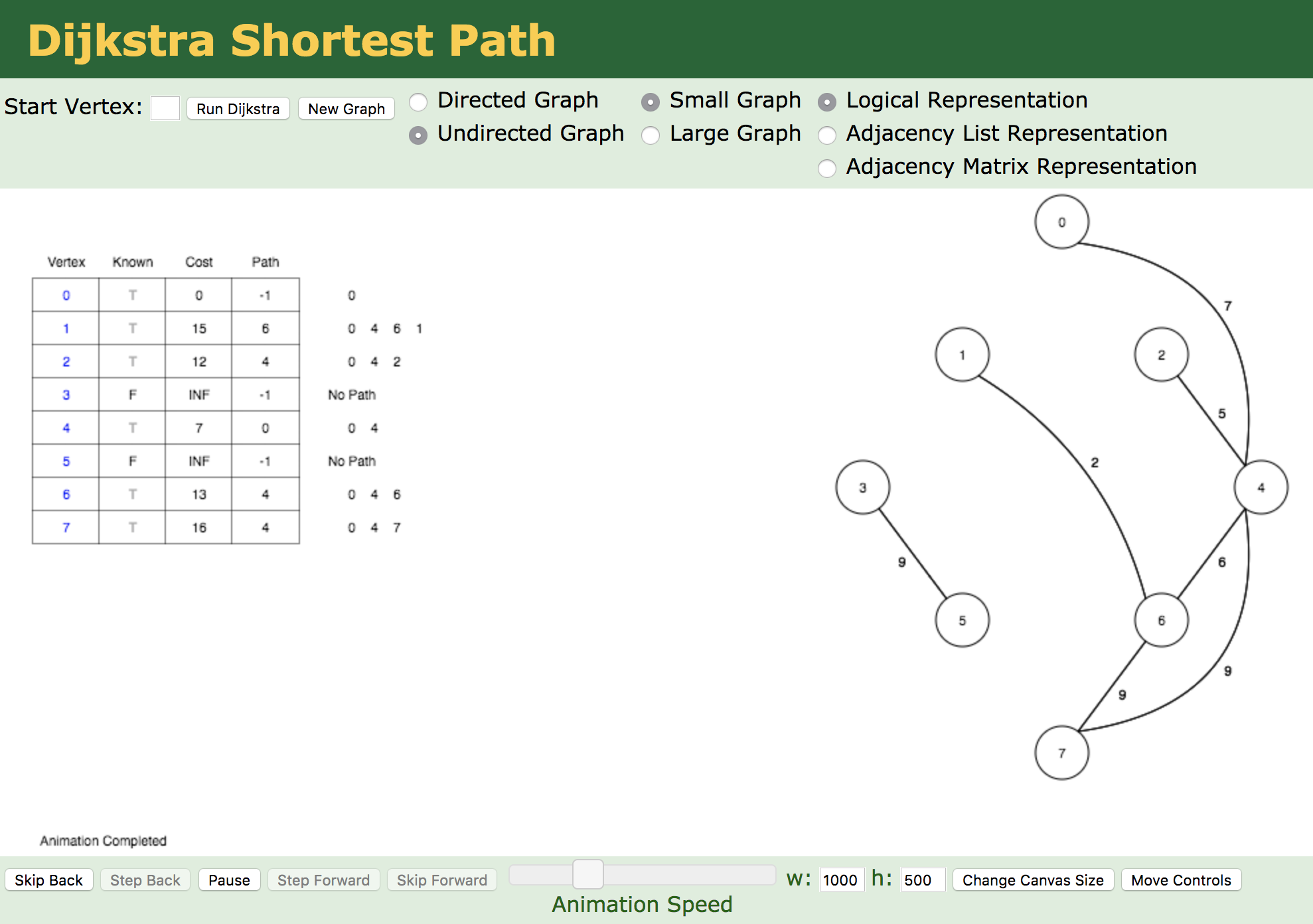
The good wise application should be findable and available. To be findable, it should be always be returned in major search engines. Common practises exist to ensure that websites are correctly indexed by search engines. What use is an application if it’s functional and findable but not available? There needs to a strong sense of security that when users access the website there is 100% availably with no connection issues. This application should also be widely available to the audience. This will require a good hosting package that will fit this requirement.

* 1. Similar Applications

With the ubiquitous nature of Graph Algorithms, the thought would have been that are many graph algorithm teaching tools that exist. However, upon researching, there are less Graph Algorithm tools than anticipated. The various tools that already exist do not satisfy all the requirements that I wish to have in an application. With experimenting with various query search on Google, it was found that the query ‘Graph Algorithm Visualization’ search returns the most valid results about Graph Algorithm tools. Here are the relevant applications that are returned after searching.

Data Structure Visualizations by University of San Francisco:

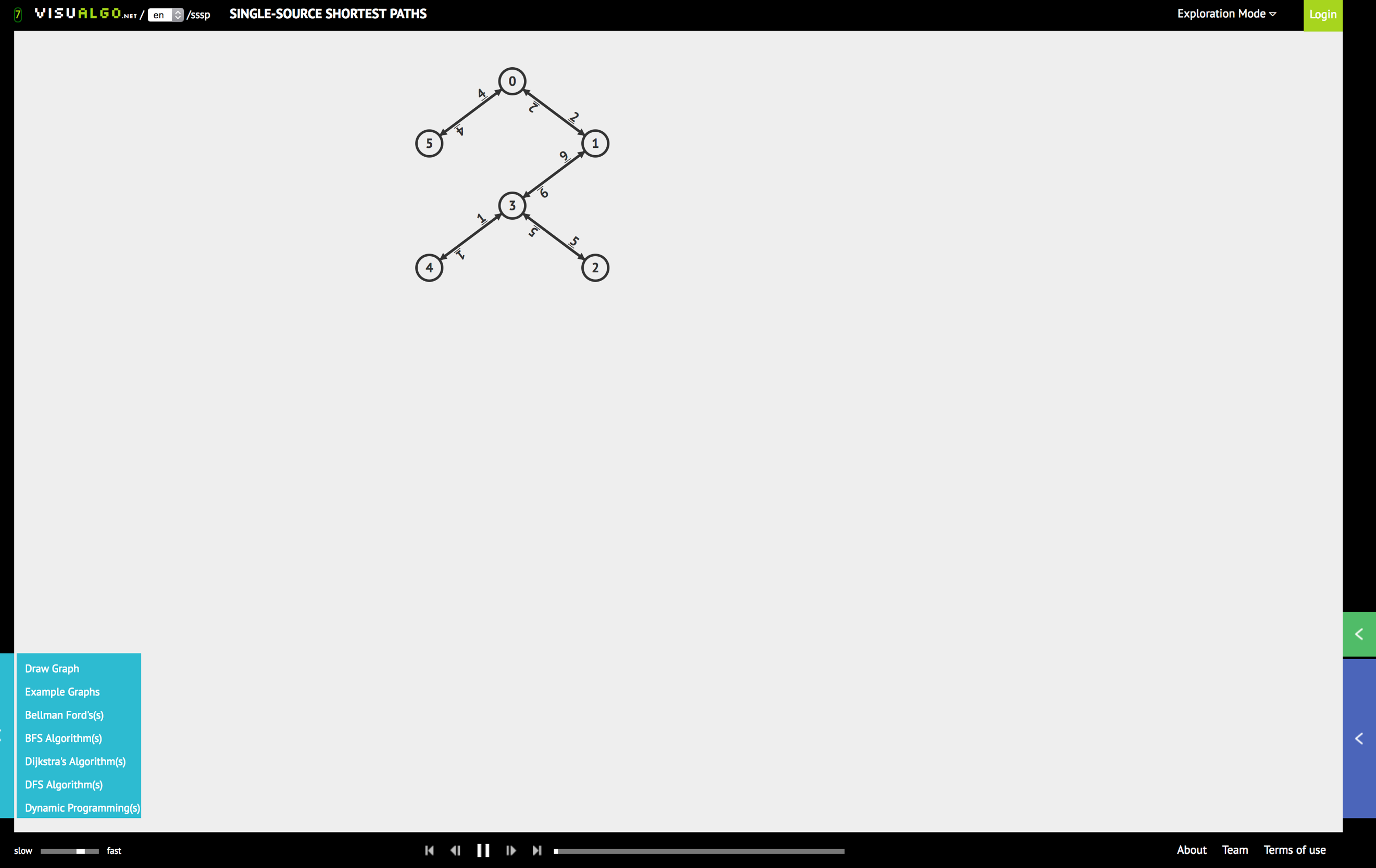
This application was the first to be returned upon searching. It is an application created by Professor David Galles from the Computer Science department in University of San Francisco. A positive is that this application does contain vast amount of amount of different algorithm support which seem to run well. However, there is not much focus on User Experience. It is not possible to change the nodes or edges on the graph in anyway. There also does not seem to be any textual information explaining what the algorithm is doing. Another issue is that even though mobile is supported, it is by no means user friendly at all as it poses difficulty to navigate the website with mobile. Also, no JSON support. This project also does not seem to be regularly maintained as the last update was pushed five years ago in 2015.



**Image of University of San Francisco Data Structure Visualization (Fig 5.0)**

Visualalgo.net

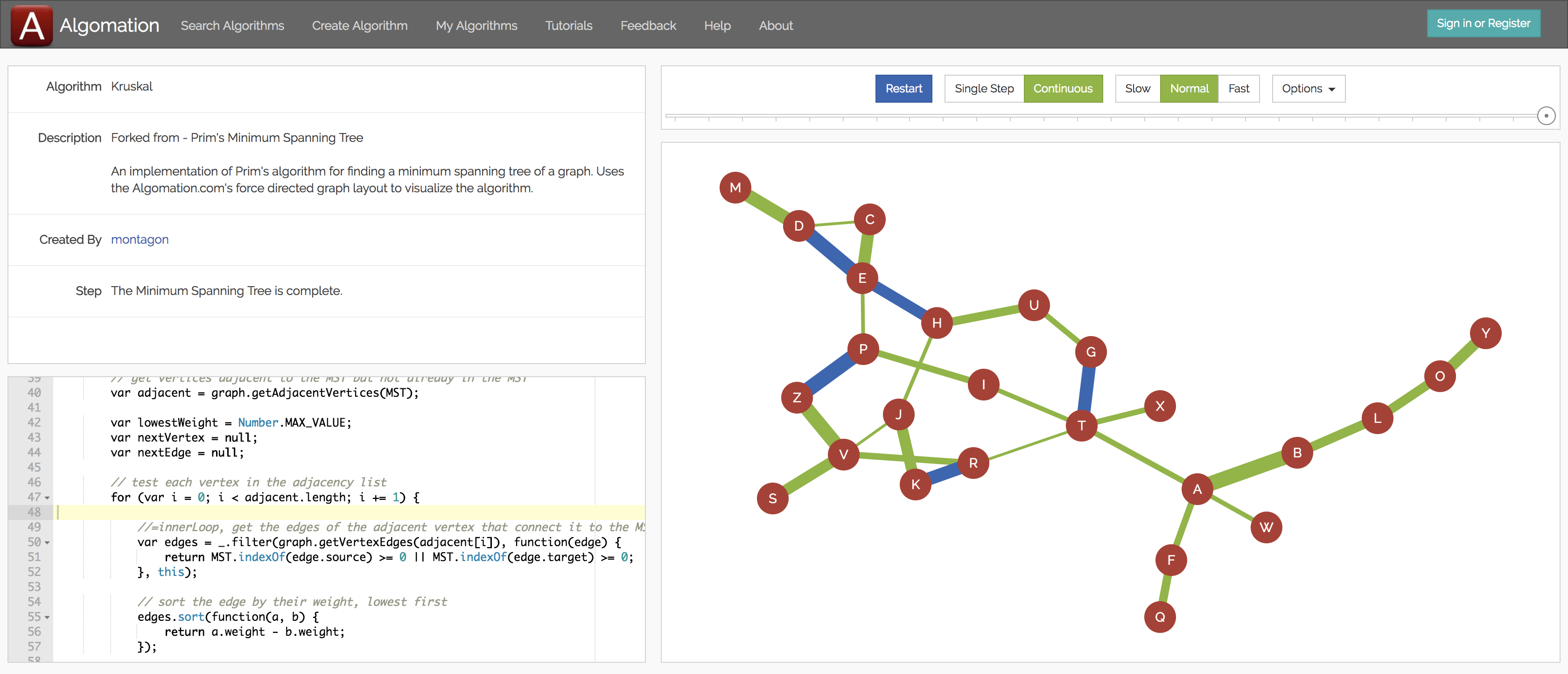
This application is the second result to be returned upon searching on Google. The application has been developed by a team of various individuals ranging from various lectures and students. There is even an accreditation from a Senior Engineer from Google. This shows that the application has had a lot of experience on to it. From the first review it does seem to be a comprehensive application with rich aesthetic algorithm visualisations. There are many positives with this application. Firstly, it has a rich User-Friendly interface with instructions on how to use the application and the algorithm. It also gives instructions on what is happening with the execution of the algorithm. However, there are aspects that I would like to improve on with my application. It seems very troublesome to easily add nodes and edges. As to do this the user has to refer to a second window supposed to just instant ‘click to add’ feature. It is also not possible for the user to edit the nodes after setting. For example, if the user wanted to change the location of a node or remove a node the user would have to load up the popup window again. Another feature that can be improved is that although the user can see textual information about what is happening, to see previous text the user would have to rewind. The the user cannot ‘pause and review’. Finally, no JSON supported in this application so it’s not easy to save and load graphs.



**Image of Visualalgo.net (Fig 6.0)**

Algomation.com

The last relevant website returned. This is developed by a Team led by an individual Duncan Meech. Similar to the others, a positive of this application is that is has an extensive list of algorithms that are supported by the application. However, like the other applications, there are disadvantages too. It has no JSON, no editable nodes, no instructions that explain what is happening in the process. Another positive that is to be taken from this application is that it has additional options to see the source code that is implemented in Scala and the ability to share the algorithm with others. Bullet points + conclusion



**Image of Algomation.com (Fig 7.0)**

Summary of current Graph Algorithm Teaching Tools

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Applications** | **Dijkstra, Kruskal, Prim support** | **Add/Remove Node and Edges on click** | **‘Draggable’ Nodes** | **JSON** | **Instructional**  **Guide on how to use website** | **Algorithm Step-by-step Instructions** | **Other notable algorithms** | **Stop**  **Rewind**  **Pause** | **Mobile**  **Support** |
| Tool by San Francisco | ✓ |  |  |  | ✓ |  | ✓ | ✓ | ✓ |
| VisualAlgo.net | ✓ |  |  |  | ✓ | ✓ | ✓ | ✓ |  |
| Algomation.com | ✓ |  |  |  |  |  | ✓ |  |  |
| My Application | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |  | ✓ | ✓ |

Specification and Design

*This chapter will expose the reader to the various specification and design decisions that were executed for this project. This will also give the reader what will be created in the application and how it is executed.*

Technological Architecture

The sub-chapter will communicate to the user about the high level technical decisions that were made in this application.

The first technical decision that was decided in this application was to determine whether this application should go along the path of being a Web Application or Desktop Application. There are various factors on what will be a more suitable approach for this application. An overview comparing the two.

Desktop Application Advantages

* Does not need Internet Connection to access the application.
* Reduction of Hosting and Maintenance costs. (Hostname, Server costs)
* Users are more ‘familiar’ to desktops applications as they have conceptual integrity.

Web Application Advantages

* Download and Installation labour not required.
* Updates and Bug fixes will be executed ‘behind the scenes’.
* Do not have to develop a separate application for separate Operating Systems
* Mobile phone support

The decision for this project was to go with a Web Application. The justification for this are highlighted in the bullet points. With the emphasis being User Experience, it would not be desirable for users to go through the trouble of downloading and installing. If I want to share the application or show it instantly, anyone could use their web browser and instantly access instead of a download. There’s a possibility that if a user runs into issues downloading and installing the application, that may hinder the User Experience. This sentiment is enforced by the next advantage point. If there are updates and bug fixes, I would rather it be done ‘silently’ so the user is not aware. If the user has a desktop application, there is a possibility the updates are prone to be intrusive. A strong feature that I would like to implement into the application is Mobile functionality. With a desktop application, this would be infeasible to do. So, this is one of the strong reasons for going for web applications.

The next design decision that the project will make is the technology stack that will be used to sculpt the web application. The main languages and tools are as follow:

Main Technologies

HTML5

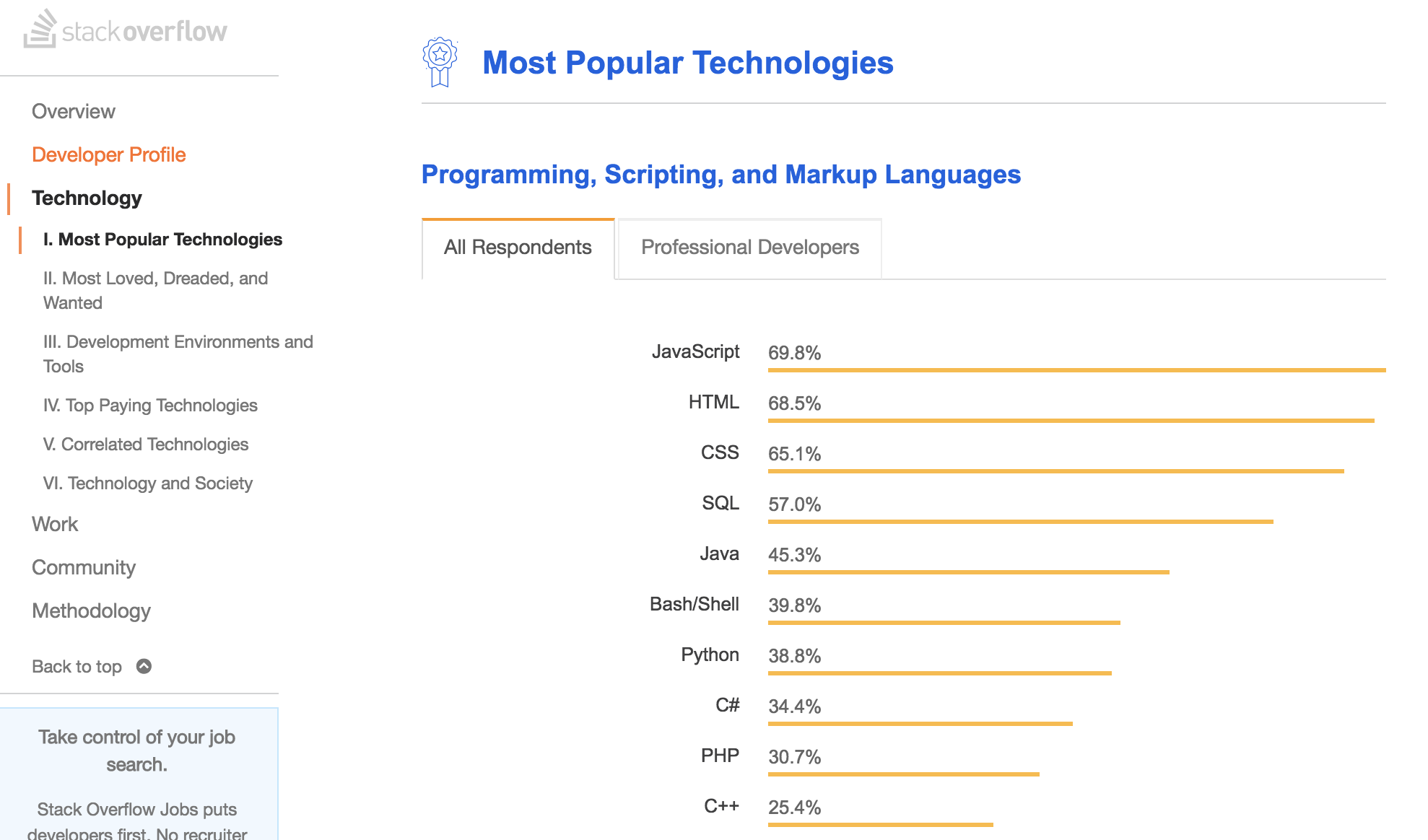
HyperText Markup Language (HTML) is the main Markup language used for structuring and presenting web pages. This markup language is what dominates the web with most web application incorporating it. With HTML5 there are various new features that have been introduced that are positives to the application. It also has improvements for mobile implementations as it is designed with less powerful devices.

CSS 3

Cascading Style Sheets is a style sheet language responsible for describing the presentation of a document that has been written in a markup language. One of the main advantages of CSS is that there is a separation of concern. It is possible to encapsulate all presentation implementation to a separate CSS file. For this application, presentation is a big concern. Having the two separate is a good design decision for this project. CSS has advanced in the way it works too, so there are more ways to create aesthetics on the application such as animations.

JavaScript

JavaScript is a high-level programming language that is one of the core technologies used for websites in the World Wide Web. All notable Web Browsers supports JavaScript and is one of the most popular languages out right now. StackOverflow which is a popular website among the technology industry did a poll with the result of JavaScript being the most popular language in 2018. [8] For the application JavaScript will be a good choice due to the numerous support that can be accessed with it. Reasons why JavaScript is so powerful is that JavaScript is the only main choice that a developer will have if developing a website. It is popular so there’s support. It is easy to pick up and you can make good JavaScript front and backend.



JQuery

With JavaScript, in the application there will be support for JQuery. The main purpose of JQuery is to simplify the front-end HTML/JavaScript. JQuery comes with syntax that make it easier to create front end scripting. Useful features that JQuery contains that will influence my application are: DOM element selection and manipulation, Effects and animations and JSON parsing.

Minimum System Requirements

For this application, the minimum system requirements that will be aimed for are any operating system support however with Google Chrome 6.0+, Internet Explorer 6.0 and Firefox 6.0+. The application will be handled so it can be tested on those. Also mobile phone support will be included to support this.

Jasmine – Test Framework

Jasmine is a popular open source testing framework for JavaScript. It was released September 2010 and gets constant support and updates from Pivotal Labs. Jasmine main concept is that it’s Behaviour Driven. In Software Engineering, Behaviour-Driven is the concept of not testing the actual implementation of a function but testing the behaviour of such function. This has the advantage of that if I update a function, regardless of the inputs and outputs, the testing can stay the same. This is important for the type of project that I have as there may be many change. Another advantage of Jasmine is that it does not rely on browsers, DOM or JavaScript framework. [9] This makes it a perfect tool to use for front-end website testing. Another advantage is made to be very easy to read. So if another eyes want to see how the tests and functionality works, then that is doable.

**function** helloWorld() {

**return** 'Hello world!';

}

describe('Hello world', **function**() {

it('says hello', **function**() {

expect(helloWorld()).toEqual('Hello world!');

});

});

Here is an example of Jasmine in action. With a simple helloWorld() function, it uses adequate behavioural practises to test the function.

JSON

JavaScript Object Notation (JSON) is an open standard format that uses human readable text to transmit data. It is a highly popular data format that is used for communication in and between web applications. With this application, JSON will be used in this application very widely in this project as a means of representing graph data. Example JSON syntax here:

var person = {"name":"John",

"age":31,

"city":"New York"

};

Graph Execution high level design

User selects desired algorithm to run and inputs source if applicable.

Application calculates the result of the algorithm

User creates graph by adding nodes and edges via mouse click or import with JSON

Verification checker

Result of algorithm shown with updated textual information

The high level engineered design is simple enough.

Step 1 – Create or Import Graph

The user has two ways to create a graph with this application. The first and common way will be using the mouse functionality. That will be adding nodes on a graph by simply left clicking on the screen. Nodes are added based on the position of the mouse cursor and users have unlimited clicks they can add nodes on the screen. Similarly, a user has the ability to remove nodes off the screen too by right clicking too. For every on node click action that happens, there is a background verification to check that there is not a duplicate node at the same position, if there is, the program will throw a helpful error. Nodes are also ‘draggable’ so users have the option of holding down left click and moving the node to their it’s desired position.

Now to add edges, nodes have two states. ‘unselected’ and ‘selected’. For a user to add an edge, there firstly must be more than one node added on the graph. After this a user can ‘click left click’ on the first node to change it state to ‘selected’. The same operation can be applied to the second/target node to successfully create an edge between them. Similarly, to the node operation, edges can also be removed from the graph by left clicking.

Secondly, a user can also create a graph with JSON. On the settings toolbar there is a button responsible for ‘Import Graph’. On click, there will be an input box that will be responsible for taking a JSON input of the graph. With a valid Graph JSON input the application will convert the JSON representation of the graph into nodes and edges. This feature are for individuals who want to save their graph for later. They have the ability to get it back.

Create Random Graph

Step 2 – Verification Checker

There are various verification checks that will be executed. The first check to see if there are any valid nodes and edges on the graph. If there are not, the application will throw a popup error to the user. The next check will just be seeing that if there’s a node id that is added is even valid. For example, there may be an input of id 5 but that is not contained in the graph. This will throw an error. This will also check that the input is valid. So, there will be no characters such as symbols or numbers that will be input there too.

Step 3 – Execute Graph Algorithm

The user chooses what graph algorithm that they want to execute. If the user wants to use Dijkstra’s or Prim’s algorithm, there will be an ID insertion too. The node Id is just what is there to symbolise the node. It looks like this:



The user will put this in. What will happen after is that with the algorithm selected. The algorithm will be ran and will output the path or sequence.

Step 4 – Visual Output

After this is all done, the webpage will update with the graph and text. A lot happens so lets split this up.

Graph:

The output visualisation is split into steps.

What will happen is that the edge that is in current iteration will get highlighted. This highlight is responsible just to alert the user that the algorithm step is at. When this happens, the step with the information about the algorithm is pasted in the text output. The text output for each algorithm will be slightly different depending on what the algorithm is. What happens is that after that happens, the edge will node will turn to another colour to highlight the change.

Media Controls

After the user successfully executes the graph there are media controls that gives the user flexibility to do various operations on the graph operation. They are:

**Pause:** User can pause current execution of graph.

**Resume:** User can resume a paused execution.

**Rewind:** User can go back a previous step

**Forward:** User can go forward a step

**Restart:** User can restart and go back to the first step.

**End:** User can go all the way to the end of the execution.

The user has the option to ‘free fall’ the execution. When play is pressed, the application will go step by step in through the iteration. The default speed of this is 2500ms. However, there are additional media control options which can change this. This comes in the format of a slider where the speed can be modified. With the slowest speed being at 3 seconds to 0.5 seconds.

Text Output

There’s a text box that a user will see that will be responsible for all the textual visualisation of the output of the algorithm messages. The textbox is meant to be fully resizable, scrollable and moveable giving the user full control of it.

Graph Libraries – Sigma

Why JavaScript was a powerful language to choose as the back-end is because of the wide range of various graph libraries that supports it. (Maybe talk about the range of libraries). Out of all the various libraries that I experimented with Sigma was a good choice to pick. With picking a Graph Library, this will avoid the whole pattern of ‘rebuilding the wheel’. As it has been done before. Fear not, with using graph libraries all they do is represent a graph programmatically. What I would have to do is still clean it up and create the graph algorithms.

Enhance UX experience

Tour Guide

One goal to achieve in this application is simplicity. One of the ways to do this is with a tour guide. Upon entering, the website, the application will see if there is a new session. If there is a new session, the application will execute the tour guide. The tour guide is an interactive instructional guide which gives an overview about how the application works. It goes over the simple instructions such as add node/delete node, it will also give instructions on how to execute an algorithm.

Aesthetics

Mentioned earlier, aesthetics plays a major role in creating a rich User Experience. To give users a good experience, there will be the possibility of them being able to customize what colour they want in the application. The user has the ability to change the following:

* Background colour
* Node colour
* Edge colour
* Text helper colour

(This section can go above or below)

With that being said, there will be a default background theme that will be supplied. It was important to supply the user with the most efficient background at a first glance that will attract users attention. This is why I incorporated a visually appealing homepage. It used a library which stimulates moving nodes and edges with interactivity. To test if my target audience enjoyed this, I went to an active Computer Science lab and ask students there what they think about the look. I ask around 15 students that were specifically second years and asked them what they think about it. This received generally positive reviews. In terms of the colours of the actual website I experimented with various colours too. A look that I was in strong flavour for was creating a dark themed website. As this promotes a rich looking application. With that background I decided that a good violet should be applied with this so that is what the default node and edge colour will look like too.

Bootstrap

Bootstrap is an open-source front-end framework that is used for creating web applications. Pioneered by engineers at Twitter, it is a toolkit that provides design templates for elements such as buttons, tables, forms etc. Bootstrap is a great addition to the project as it comes with Responsive Web Design which is essential for the vision of extending the application to smaller devices such as mobile phones and tablets. It is a very easy framework to implement into the application with its comprehensive documentation and endless support on StackOverflow. Another advantage of this framework is the Browser Compatibility. There does not be an issue about an aspect of the application looking different or behaving different on different as bootstrap will help with this issue.

Git & GitHub

Git is a distributed version control system. Its purpose is keeping track of changes to project files. Another tool which a benefit to the application it’s will be as free, very fast to operate. It comes with the core functionalities of being able to easily see differences to files. It also provides the safety mechanism of if files were accidently deleted or changed they can be reverted back to their previous correct state. GitHub is just the online repository that provides all the functions of Git version control. It is a way to connect with the local Git project to allow developers to save code and collaborate with other developers. There will be a GitHub account for this project with a repository made so others can see this project. This will be provided with documentation on the page so the application can be easily accessed and used.

WebStorm

Model View Controller

Reference

[1] – “Final Year Project Choices” -https://moode.essex.ac.uk/mod/data/view.php?d=296&rid=26934

[2] – “Email from Sanderson Michael”

[3] – “Final Year Initial Report” Emmanuel Odame – Asante

[5] - Cormen, Thomas; Charles E Leiserson, Ronald L Rivest, Clifford Stein (2009). Introduction To Algorithms (Third ed.). MIT Press. p. 631. ISBN 0262258102.