Path Visualizer

UCS503 Software Engineering Project Report

Mid-Semester Evaluation Submitted by:

(102003417) Piyush Sharma

(102003400) Sukhvir Singh

(102003405) Sartaaj Singh

(102003168) ManinderPal Singh

BE Third Year, COE

Group No: 16

Submitted to: Kanupriya

Computer Science and Engineering Department

TIET, Patiala

September 2022

TABLE OF CONTENTS

1.Project Selection Phase

Software Bid/ Project Teams
UCS 503- Software Engineering Lab

Group : __COE16__ Date – 3/8/2022

Team Name: SENTINELS
Team ID (will be assigned by Instructor):

Please enter the names of your Preferred Team Members. :

- You are required to form **a three to four person** teams'
- Choose your team members wisely. You will not be allowed to change teams.

Name	Roll No	Project Experience	Programming Language used	Signature
Piyush Sharma	102003417	Data Analysis of AirBnb(DBMS), Tweet Sentiment Analysis(AI), Food Ordering App	PostgreSQL, Python, Android Studio Development.	
Sukhvir Singh	102003400	Courier Management System(DBMS), Speech to Text(AI)	MySQL, Python	
Sartaaj Singh	102003405	Data Analysis of Airbnb (DBMS), Speech to Text (AI)	MySQL, Python	
ManinderPal Singh	102003168			

Programming Language / Environment Experience

List the languages you are most comfortable developing in, **as a team**, in your order of preference. Many of the projects involve Java or C/C++ programming.

1. Python
2. C/C++
3. MySQL and PostgreSQL

Choices of Projects:

Please select **4 projects** your team would like to work on, by order of preference: *[Write at-least one paragraph for each choice (motivation, reason for choice, feasibility analysis, etc.)]*

First Choice	**Smart Attendance System Using Face Recognition.** This project will be useful in many colleges which can prevent proxy attendance in class system. It also helps us to implement some Machine learning in the project itself, which is our strong motive to do this project. We will use database to store the attendance of the students.
Second Choice	**Bug Tracking System.** Our motivation for doing this project is complexity and handling of databases more than any other projects in this list. This system will consists of an issue, priority wise it will be handled by different users. This project has many functions, Ex- LoginPage, RegisterPage, Create Project, Create Bug, and Database system to store all these bugs and users.
Third Choice	**Path Visualizer**. It is a visualize project, we use various Path Algorithms such as Dijkastra's Algorihthm, DFS, BFS and A* search to visualize shortest path between START and GOAL, we need to learn and implement CSS and Javascript (ES6) to make this project.. This project will have multiple functionalities too.
Fourth Choice	**Sorting Algorithms Visualizer.** It is a visualize project, we use sorting algorithms such as Merge Sort, Quick Sort, Bubble sort algos to visualize how much time sorting algos take to sort the array of various sizes, to implement this idea we need a strong foundation of CSS and Javascript..

Additional Remarks/ Inputs

Please tell us about any other factors that we should take into consideration (e.g., if you really would like to work on a project for some particularly convincing reason).

2. Planning Phase

Project writeup

PATH VISUALIZER

The goal of this project is to create a web based e-Learning tool, 'Path Visualizer', which can be used to visualize shortest path algorithms. The conceptual application of the project is illustrated by implementation of algorithms like Dijkstra's , A* and DFS. A visual pathfinding program that allows the user to create their own obstacles or mazes and then run different pathfinding algorithms on it.

It is a more practical variant on solving mazes and riddles. It is used in finding directions between physical locations. This is the most common usage, and web mapping tools such as Google Maps use the shortest path algorithm, or a variant of it, to provide driving directions. Social networks can use the algorithm to find the degrees of separation between people.

This field of research is based heavily on different path algorithms for finding the shortest path on a weighted graph.

Our website is a visual pathfinding program that allows the user to create their own obstacles or mazes and then run different pathfinding algorithms on it.

Features: -

i. Tutorial:

In this user can learn how to use the visualizer website. How they can set the Weight and Wall node, and visualize the Dijkastra's Algorithm.

ii. Set Start and Goal Node:

Start Node and Goal Node are the most vital parts of path visualizing concept, user can place them anywhere on grid and get a better understanding of the concepts.

iii. Show Code

The user can view code and the working of path visualizer simultaneously so that the user can keep up the track according to the code and can have better understanding.

iv. Add Wall

To increase the difficulty, the user can add wall node which will act as a resistance on grid so that the user can be aware of the concept of different paths more deeply and easily. This is implemented as wall node in this project

v. Add Weight

To have a more clear understanding about the resistance and time, the user can add weight node which will also act as a resistance on grid so that the user can be aware of the concept of different paths more deeply and easily.

vi. Visualize

After setting start and goal node along with adding resistance, the grid setup is complete and the user can start the visualizing the shortest path by visualize function.

vii. Clear Board

After visualizing any of the algorithms the user can clear the board and get in touch with any other algorithms to visualize it again.

Feasibility Report

1.Technical Feasibility

This project is does not need any implementation of databases, hence it is memory efficient. The frontend of this application is implemented using CSS, Javascript and basic level of React which would support the different graphics for grid, different animations and visualizer. Different algorithms and their implementation are stored using Javascript(ES6) only.

2.Legal

The group owns the copyright in the software that we create in the project. The group agrees to transfer the copyright to the Client and to provide the Client with unrestricted license to use the system. It is just possible that a project may develop concepts that could be patented. If such a situation arises, the group collectively owns the rights to all patents associated with the System. We understand that the use of opensource solutions is a viable option and that there are not any serious licensing issues to this extent.

3.Economical

The project aims to visualize various algorithms for better understanding and learning the concepts more easily. It is a open for all website so any student/teacher can access it as it is free of cost. Nowadays student study from book and they understand only theory or book language but they don't have practical knowledge of algorithms moreover smart classes with projectors, teacher can show our website and students will be able grasp knowledge about algorithms in a more efficient and effective way which they will not forget in their life. Its not only a benefit for students but also for teachers also.

4.Operational

The project provides an overall understanding of algorithms which students usually gets confused with. There are very less chances for errors, it is a very simple and easy to use website. Our basic goal was to implement a path visualizer so that with more interaction the students understand path finding algorithms with better clarity. Successful implementation of the system will enable flexible choosing of algorithms, which will enhance the ability to understand algorithms more effectively.

.

Gantt Chart

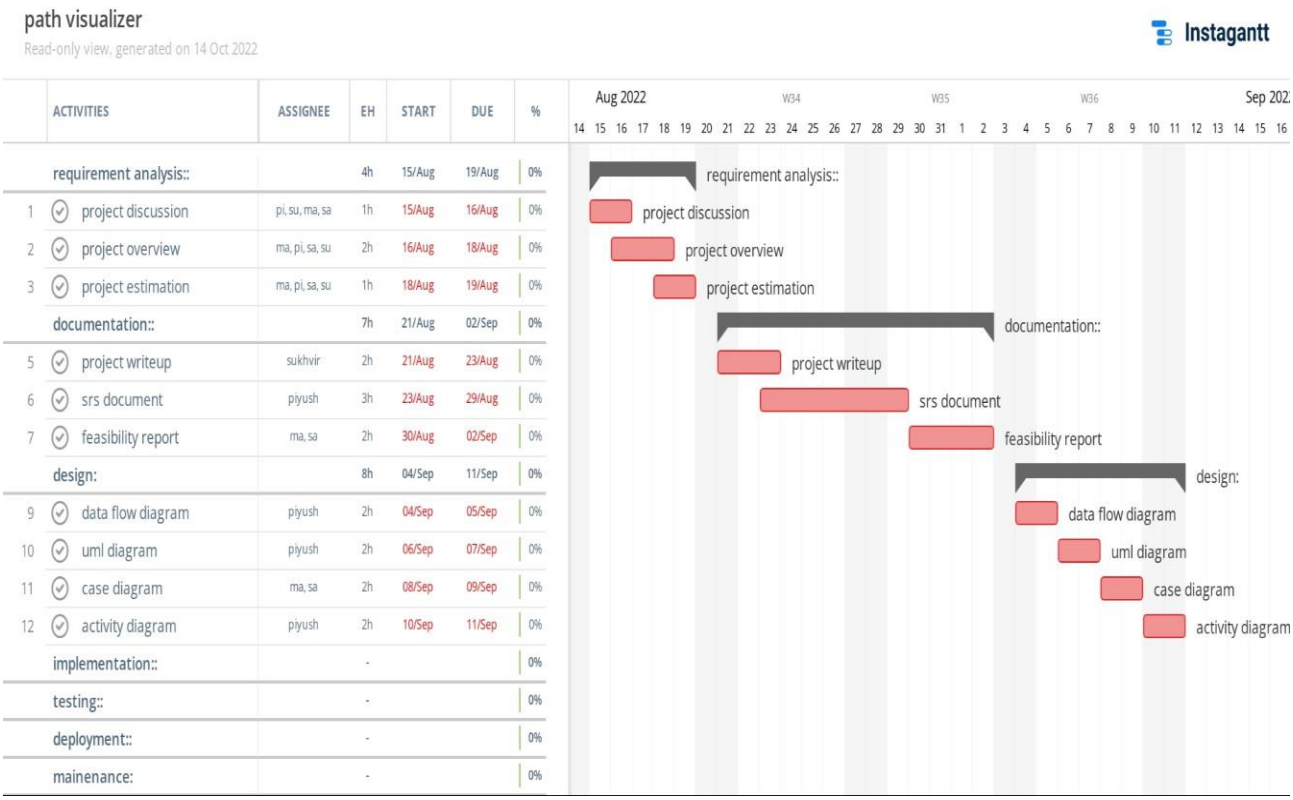

Fig1. Gantt Chart

3. Analysis Phase

Use Case Diagram

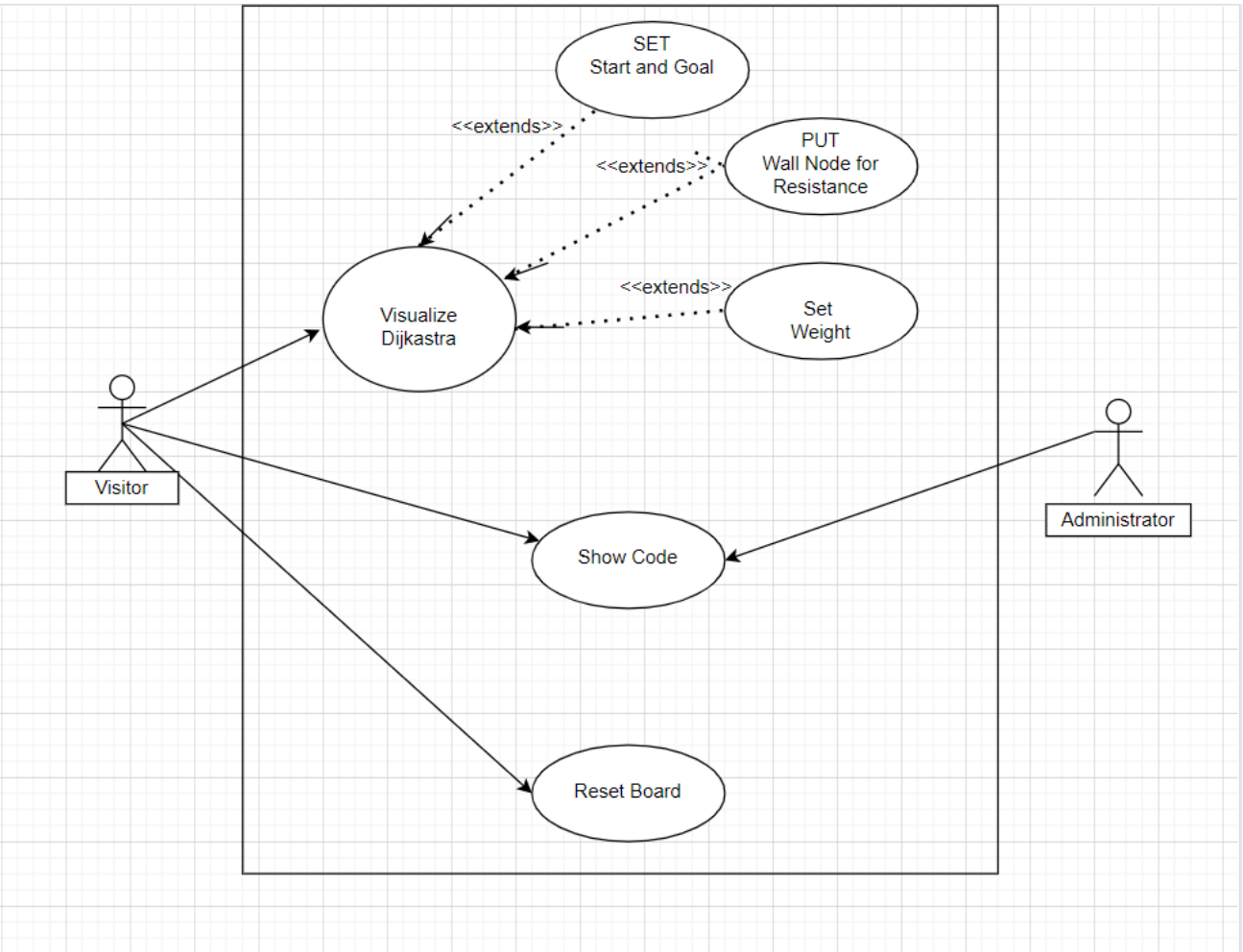

Fig2. Use Case Diagram

Use case Scenario

1.Use Case Title	Tutorial
2.Abbreviated Title	Tutorial
3.Use Case ID	8
4.Actors	User
5.Description It appears before using path visualizer	
5.1 Preconditions Just wait it will show automatically	
5.2 Task Sequence Open website	
5.3 Post Conditions Click anywhere to continue further in path visualizer	

1.Use Case Title	SET Start Node
2.Abbreviated Title	SET Start Node
3.Use Case ID	1
4.Actors	User

5.Description

User can set the start node anywhere on the board

5.1 Preconditions

Open the website

5.2 Task Sequence

Hold the start node and place it anywhere you want

5.3 Post Conditions

Start node is set and now go set goal node also

1.Use Case Title	SET Target Node
2.Abbreviated Title	SET Target Node
3.Use Case ID	2
4.Actors	User

5.Description

User can set the Target node anywhere on the board

5.1 Preconditions

1.Open the website

2.Start node should be set

5.2 Task Sequence

Hold the target node and place it anywhere you want

5.3 Post Conditions

Target node is set and now go add resistance

1.Use Case Title	Wall Node
2.Abbreviated Title	Wall Node
3.Use Case ID	3
4.Actors	User

5.Description

User can set resistance anywhere on the board except on start and target node

5.1 Preconditions

1.Open the website

2.Start and target node should be set

5.2 Task Sequence

Click on boxes on board and resistance will be added

5.3 Post Conditions

Resistance is added and now go choose desired algorithm

1.Use Case Title	Set Weight
2.Abbreviated Title	Set Weight
3.Use Case ID	4
4.Actors	User

5.Description

User can choose weight .

5.1 Preconditions

1. Start node should be set

2.Resistance should be added(if wanted)

5.2 Task Sequence

Click on any number of weight.

5.3 Post Conditions

Weight is chosen, now visualize path

1.Use Case Title	Visualize
2.Abbreviated Title	Visualize
3.Use Case ID	5
4.Actors	User

5.Description

Just click on visualize button

5.1 Preconditions

1. Start and target node should be set

2. Algorithm must be chosen

5.2 Task Sequence

Click on visualize button

5.3 Post Conditions

After visualization, user can clear board or can see code

1.Use Case Title	Reset Board
2.Abbreviated Title	Reset Board
3.Use Case ID	6
4.Actors	User

5.Description

Just click on Clear board button

5.1 Preconditions

1. Start and target node should be set
2. Algorithm must be chosen
3. Path should be visualized

5.2 Task Sequence

Click on Reset board button

5.3 Post Conditions

After reset board, user can choose any other algorithm or can just end

1.Use Case Title	Show Code
2.Abbreviated Title	Show Code
3.Use Case ID	7
4.Actors	User

5.Description

Just click on Clear board button

5.1 Preconditions

1. Start and target node should be set
2. Algorithm must be chosen
3. Path should be visualized

5.2 Task Sequence

Click on Reset board button

5.3 Post Conditions

After reset board, user can choose any other algorithm or can just end

Swimlane diagrams

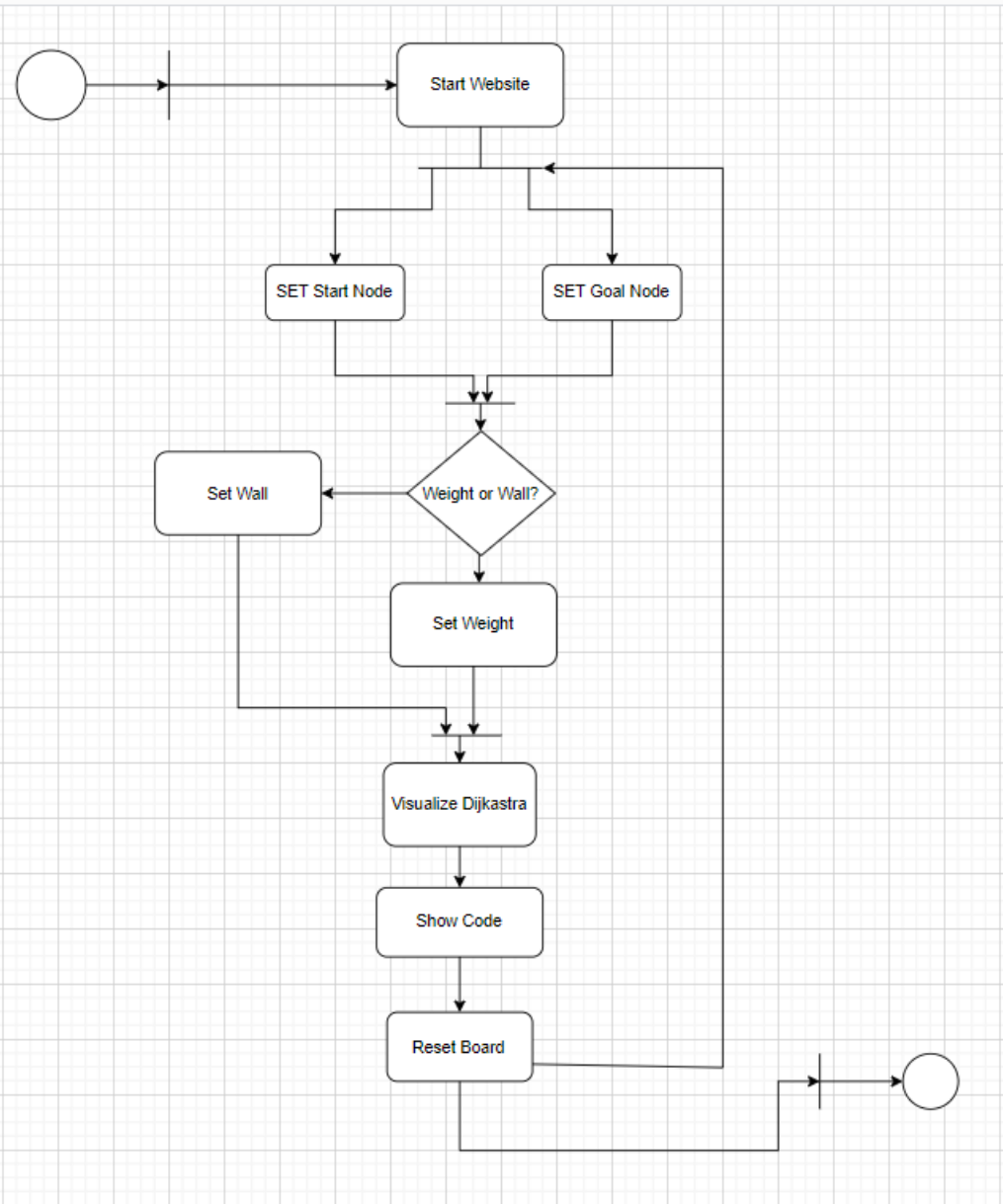

Fig3. Swimlane Diagram

Data Flow Diagrams

i) Level 0

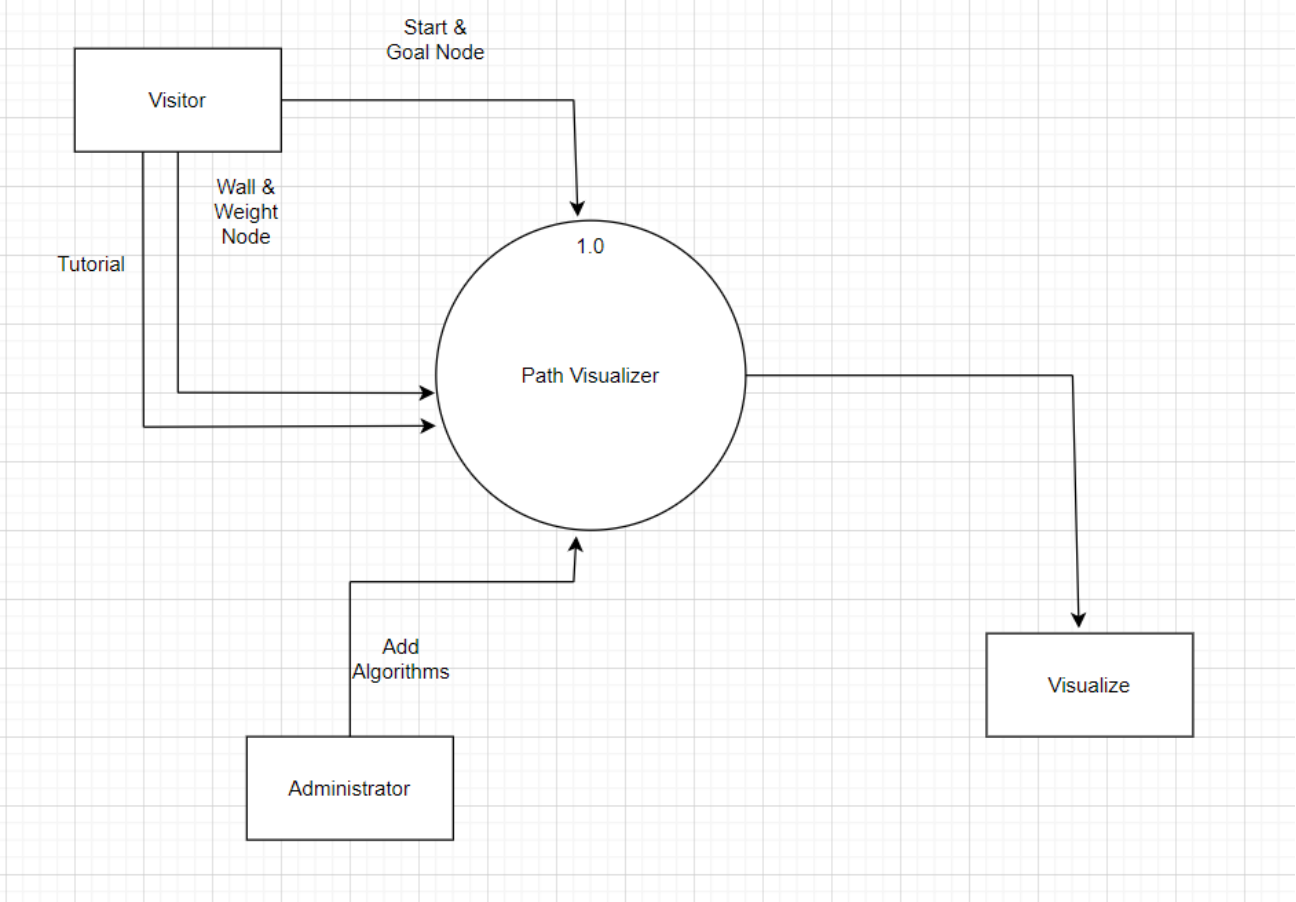

Fig4. DFD Level 0

ii) Level 1

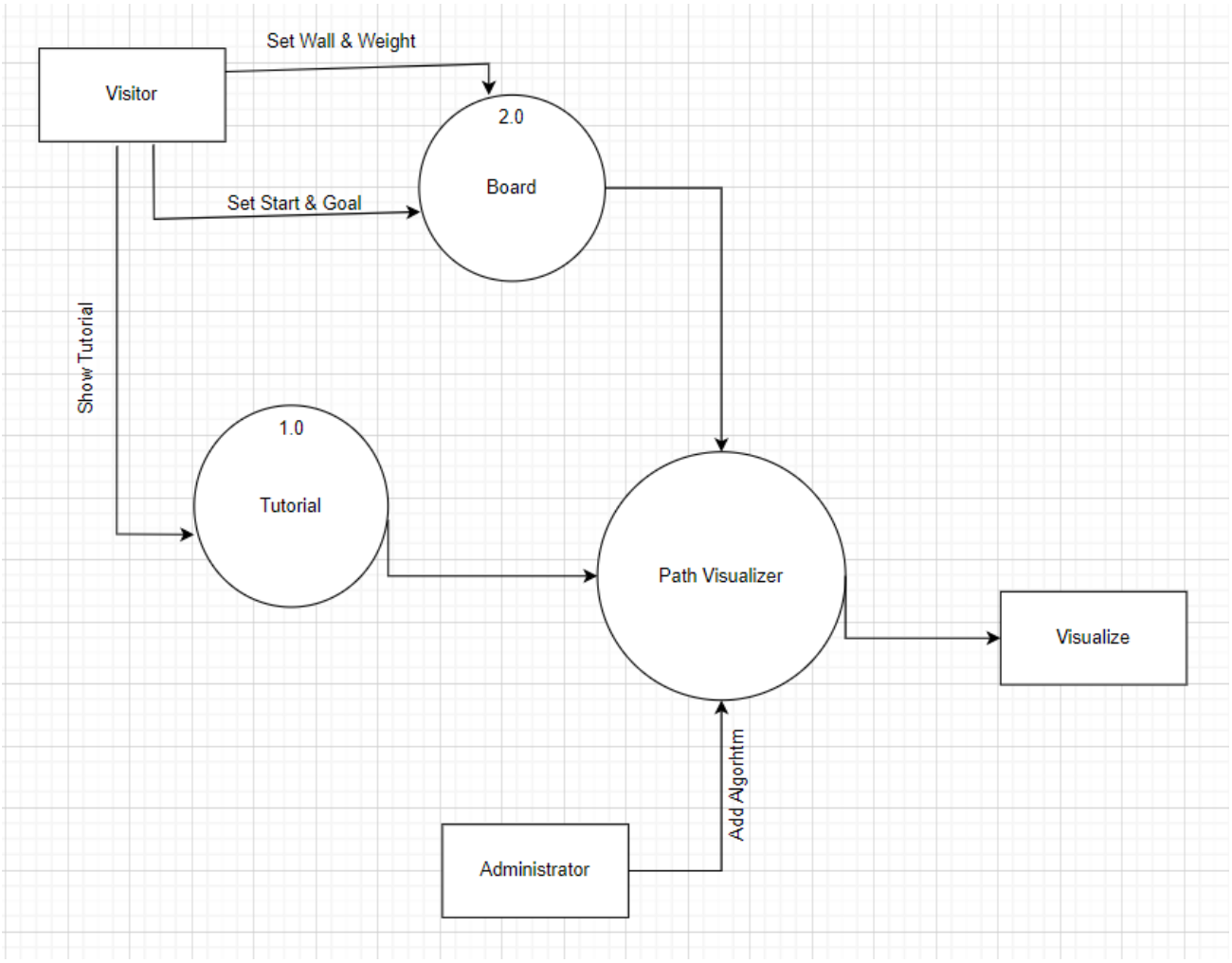

Fig5. DFD Level 1

iii) Level 2

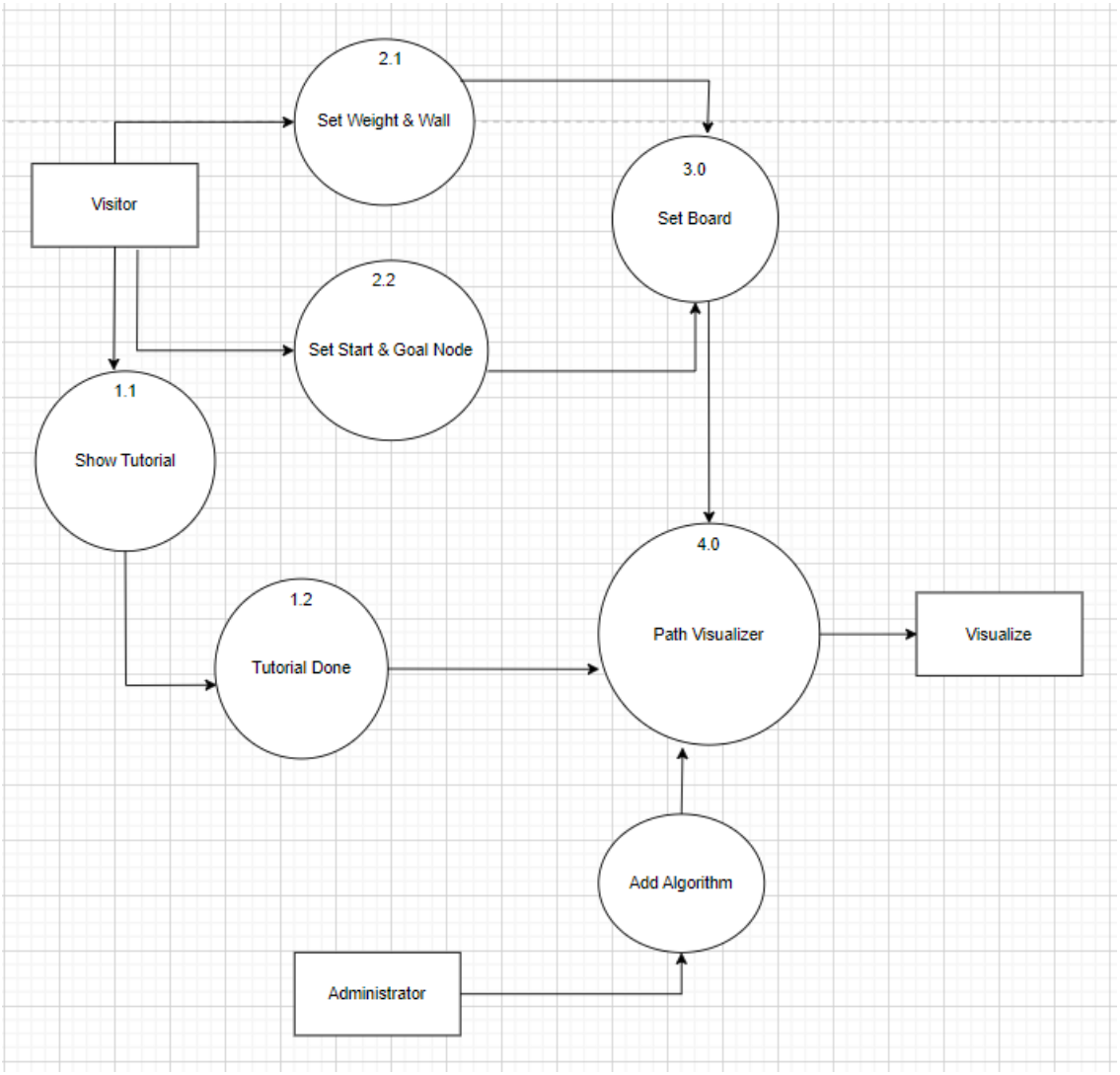

Fig6. DFD Level 2

Software Requirements Specification

for

Path Visualizer

Version 1.0 approved

Prepared by Piyush Sharma

Sentinels

3rd October 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1. Introduction

1.1 Purpose

This System Requirement Specification (SRS) aims to provide the readers and users information about the system and its functions and specifications. SRS describes the data, functional and behavioral requirements of the software. This software is designed to manage the events in the college .This will take the users requirements for about events. According to the users requirement it schedules the events and also helps in maintaining the workforce and the job roles that are assigned to them.

1.2 Document Conventions

Heading	Bold
Paragraph	Times New Roman
Heading Size	18
Content Size	12

1.3 Intended Audience and Reading Suggestions

The intended audience in this SRS are developers and coding enthusiasts, proffessors , teachers and students who want to know more about path algorithms and want to clear their concept by visualizing stuff.

1.4 Product Scope

Our website provides an interactive platform for learning various types of path algorithms like Dijkastra's Algorithm, Best First Search, A* Search, Breadth First Search and Depth First search, It can be accessible from any place. It provides full freedom to user to set START and GOAL node anywhere in the grid, the user can also add resistance to the path, which will make the learning more insightful and efficient.

1.5 References

1. https://github.com/clementmihailescu/Pathfinding-Visualizer
2. https://www.youtube.com/watch?v=msttfIHHkak

2. Overall Description

1.2 Product Perspective

The software will be a new independent product, that it is not a component of another program. All the forms used in product follows a clear and logical structure.

2.2 Product Functions

This program designed to assist proffessors teaching their students, students who want to learn interactively and other users whose line of business deals with E-learning. It shall perform the following functions:

1) The user is first familiarized with the envoriment by tutorial screen.
2) The user can set Start and Goal node anywhere on the grid.
3) The user can set Wall and Weight node for resistance in the path
4) The user can view the Dijkastra code too for better understanding.
5) The user can Visualize the shortest path by clicking on visualize button
6) The user can reset the board for changing the parameters of the grid.

2.3 User Classes and Characteristics

The goal is to design software for a E-learning company including event scheduling for different users. These user types are listed below:
1. Students
2. Proffessors

All the above listed users have different education background and expertise level in using the system.
Our goal is to develop software that is easy to use for all types of user including the security officer. The user must have the following characteristics:

1)The user must be a computer literate and has no difficulty in using smart devices.
2)The user need not be aware of internal working of the site but must be aware of the basic functionalities that the site provides.
3)The user should have basic understanding of programming

2.4 Operating Environment

Operating environment for the Path Visualizer is:

Operating system: Windows
Platform: Javascript, ReactJS

2.5 Design and Implementation Constraints

The following list presents the constraints that are imposed upon implementation and design of the Path Visualizer:

1) Due to the small form factor, only limited graphics can be supported on the display screen.
2) A general knowledge of basic computer skills is required to use the website.
3) A general knowledge of basic programming skills is required to understand the algorithms better. 4) Memory requirement is there for storing working of these algorithms

2.6 Assumptions and Dependencies

1) A general knowledge of basic computer skills is required to use the website.
2) Database have not been implemented in this.
3) Response time for loading the software and for processing a transaction should not be longer than 5 seconds

3. External Interface Requirements

3.1 User Interfaces

- Our website provides an interactive environment to the visitors who want to learn more about Dijkastra's Algorithm and wants to visually learn the whole process.
- It will have a user friendly view of the whole system with simple and easy understanding of all the operations being provided as the command buttons are functionally labelled.

3.2 Hardware Interfaces

CPU	64 bit x 86
RAM	4GB
Display	1024 x 768

3.3 Software Interfaces

- The operating system required is minimum windows 7.
- The framework that we are using is ReactJS framework
- The website requires an active internet connection for its functioning.

3.4 Communications Interfaces

No communication interfaces are needed as our system is a standalone system.

3.5 System Features

Implementation of User Interfaces

Through the use a combination of Node Modules and ReactJS Framework user interfaces were implemented.Below are a number of user interfaces that can be used by different system users.

- **Tutorial:** This is the first interface for the Path visualizer . This interfacecan be accessed by all the system users and has information about what has to be done to access the different system functionalities.

- **Set Start:** In order for a visitor to visualize the whole process, he/she need to set the start node on the grid.
- **Set Goal**: In order for a visitor to visualize the whole process, he/she need to define the target or goal node on the grid.
- **Add Wall:** In order for more interactive and clear understanding wall functionality is added to the grid.
- **Add Weight:** In order for more interactive and clear understanding weight functionality is added on the grid.
- **Reset Board:** To Visualize again with new starting and ending point reset functationlity is added.
- **Show Code:** To better understand the Dijkastra's Algorithm, show code functaionality is added for better understanding.

4. Other Nonfunctional Requirements

4.1 Performance Requirements

Although the system is simple one a literate organizer with basic computer knowledge is needed to run the system. The person who is handling the frontend should have basic understanding on React.

4.2 Safety Requirements

This project doesnot contains any database, hence there is no information stored on the website.

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: To Be Determined List

User Story Cards

User Story Card 1:

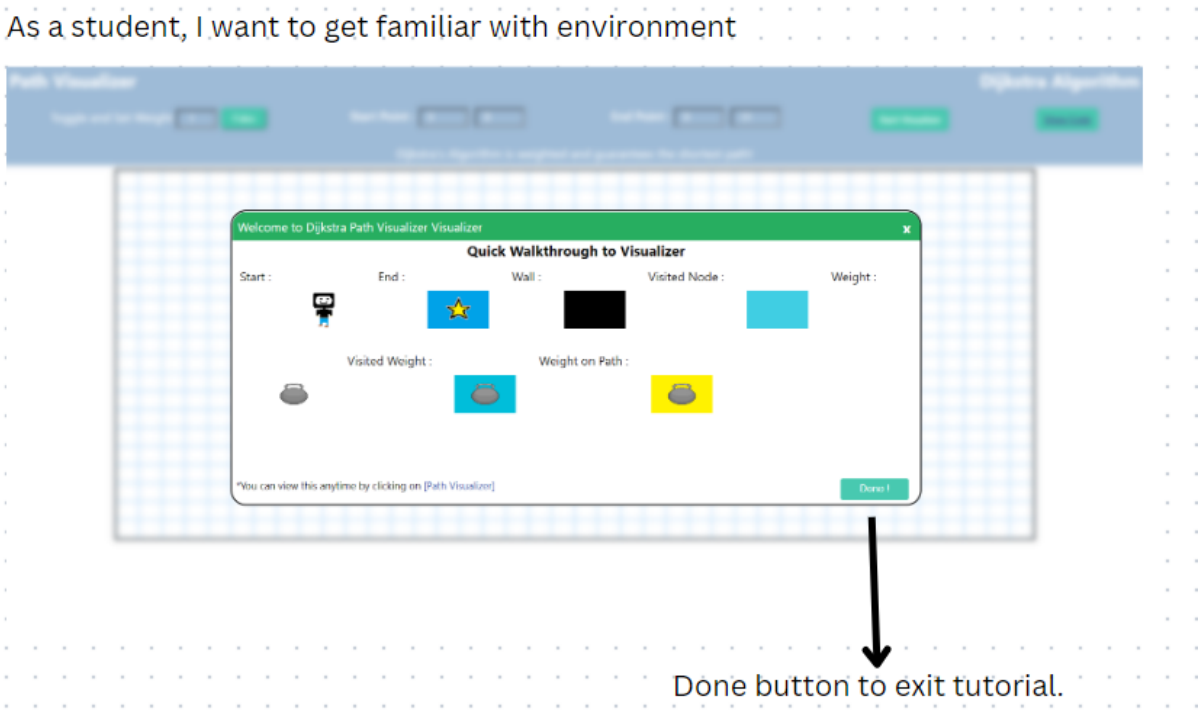

Fig7. Front Story Card 1

Back Story Card 1:

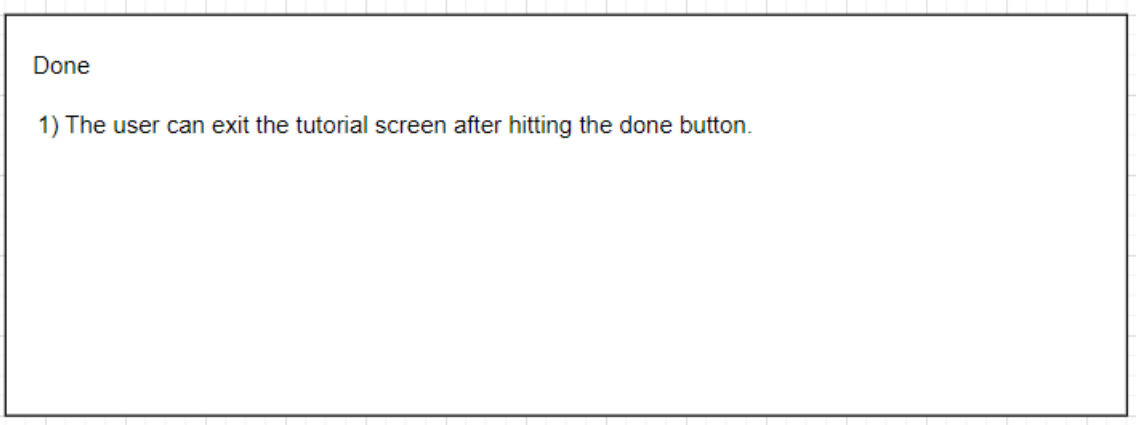

Fig8. Back Story Card 1

User Story Card 2:

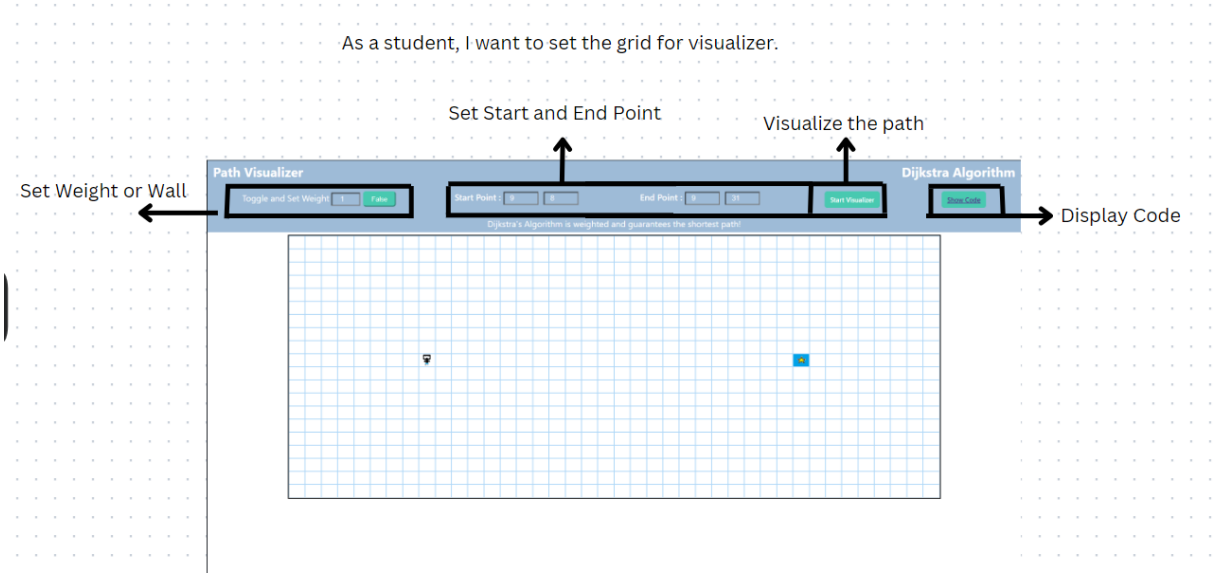

Fig9. Front Story Card 2

Back Story Card 2:

Set Wall & Weight
1) The user can toggle between wall and weight
2) The user can set the amount of weight

Start and Goal Node
1) The user can set the Start and Goal node

Visualize
1) The visitor can visualize the path using Visualize button

Show Code
1) The user can see the code of Dijkastra's algorithm

Fig10. Back Story Card 2

User Story Card 3:

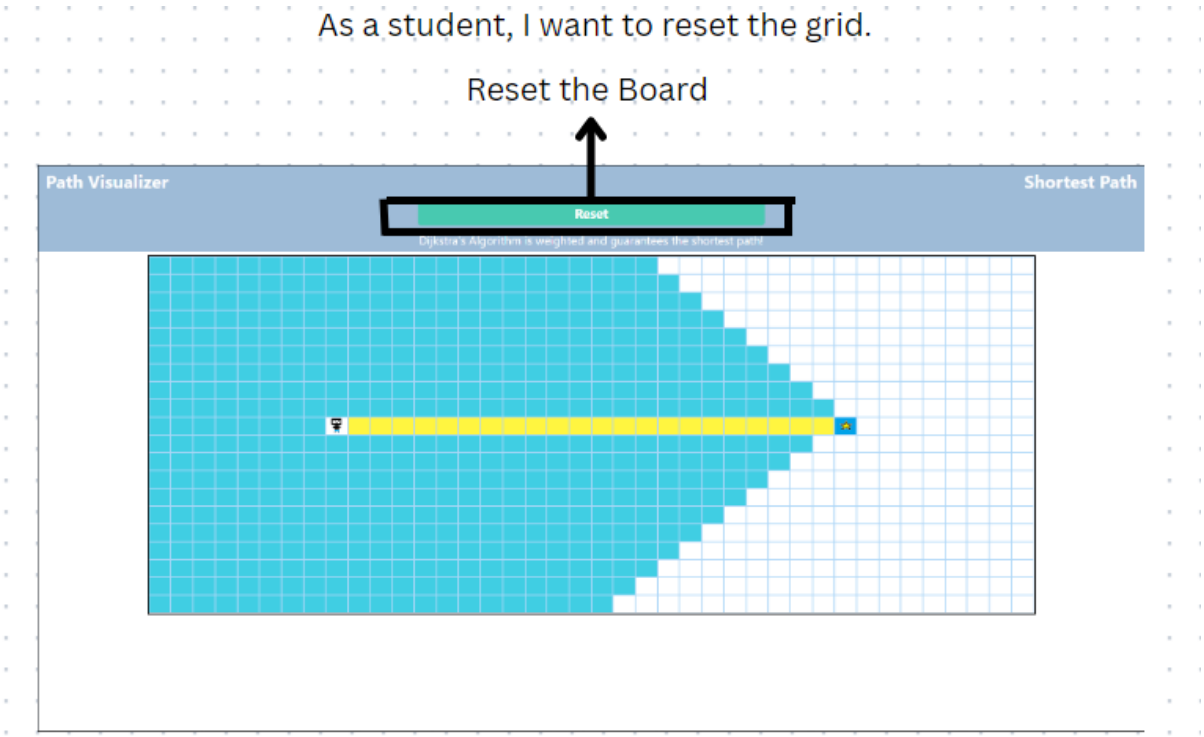

Fig11. Front Story Card 3

Back Story Card 3:

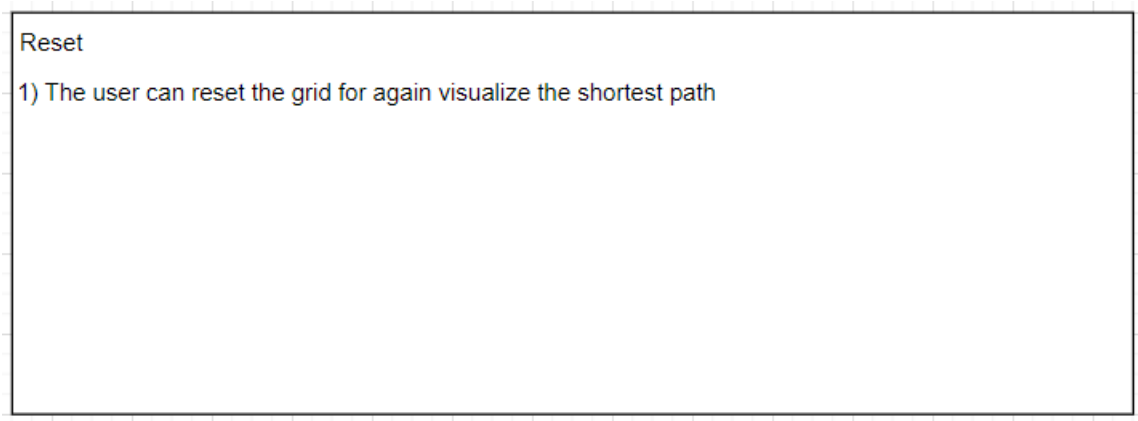

Fig12. Back Story Card 3

4. Design Phase

Class Diagram

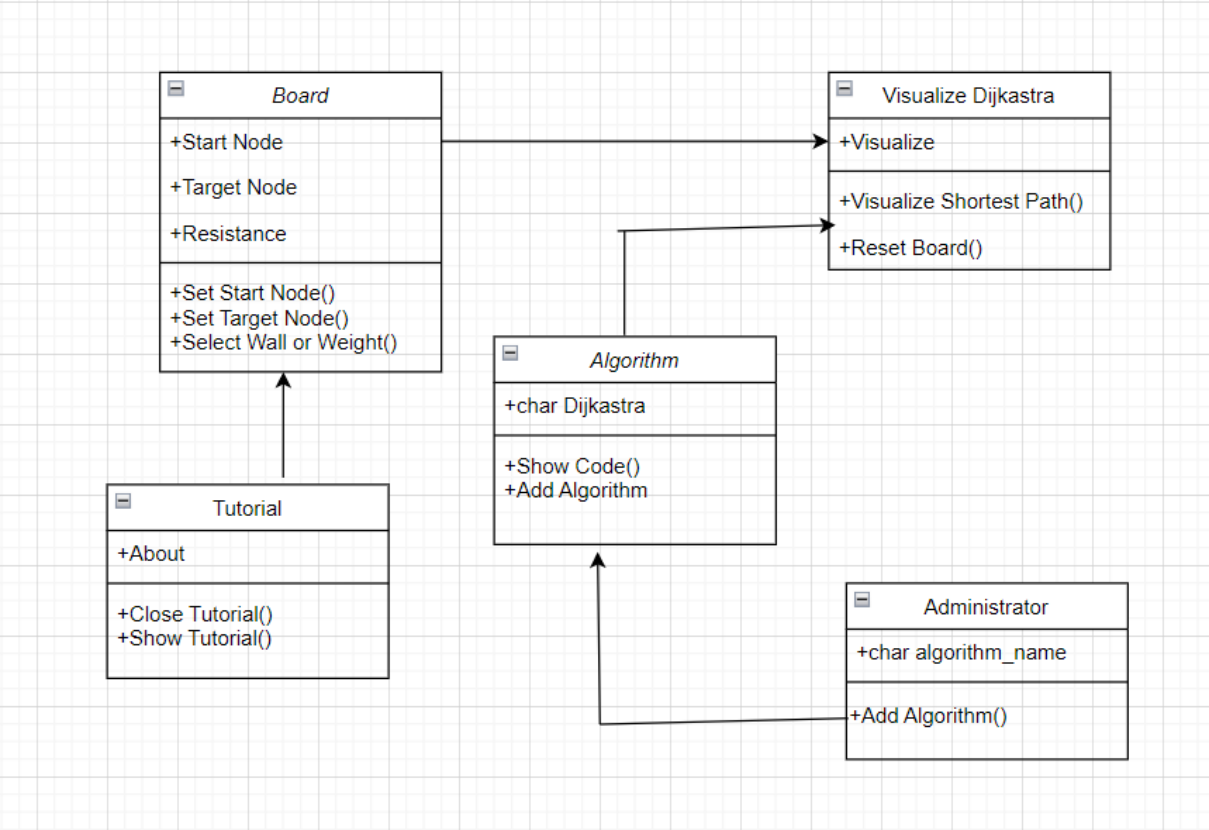

Fig13. Class Diagram of Path Visualizer

Sequence Diagram

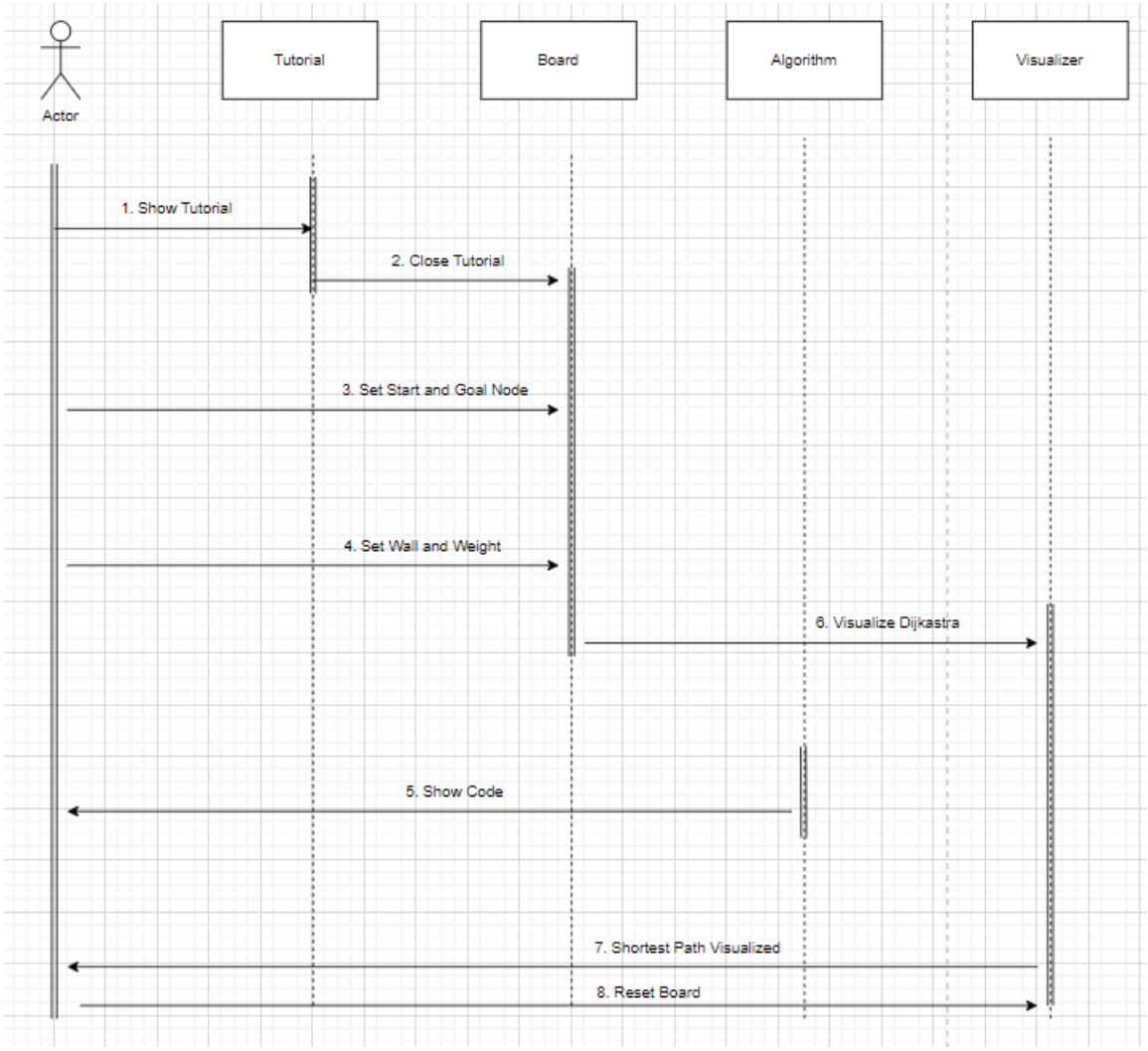

Fig14. Sequence Diagram

Collaboration Diagram

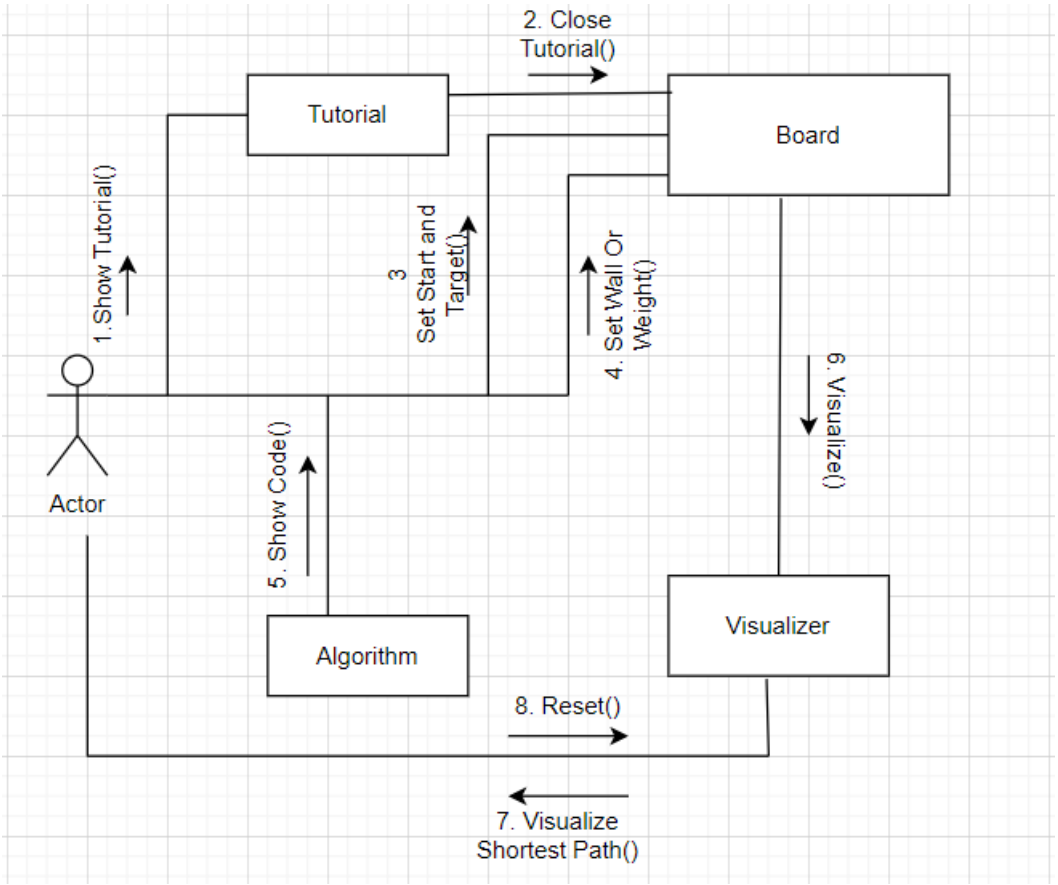

Fig15. Collaboration Diagram

State Chart Diagram

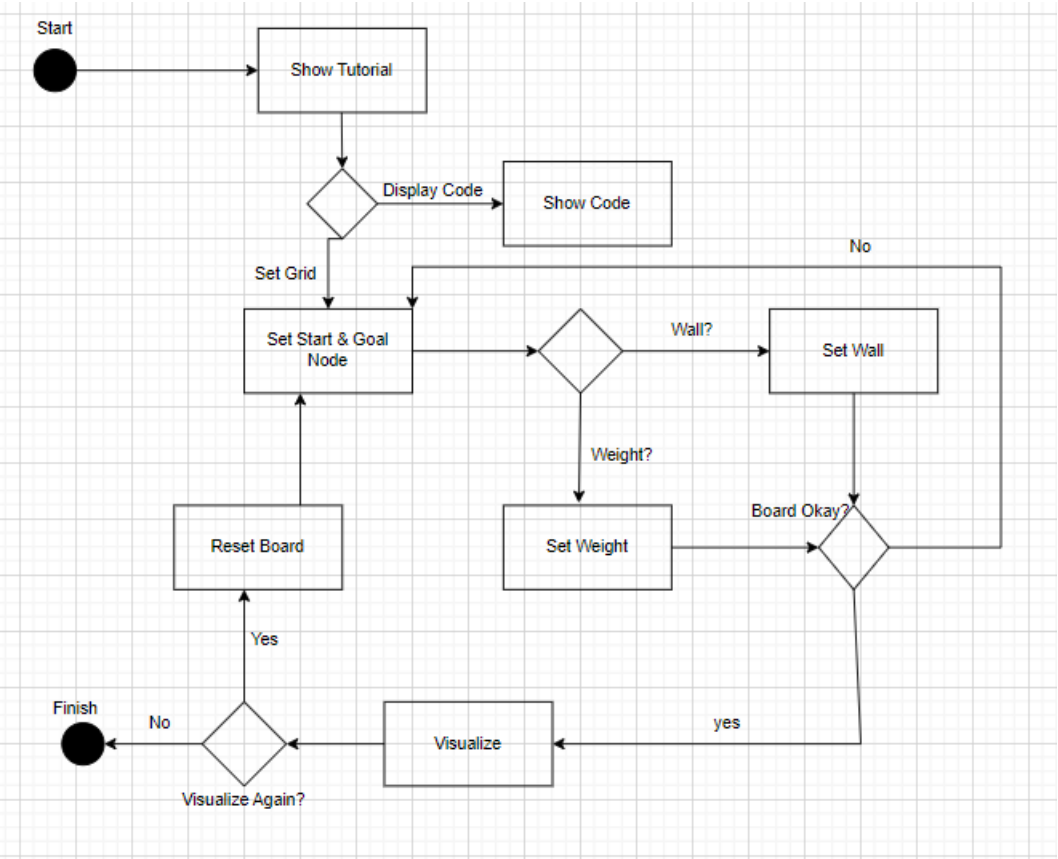

Fig16. State Chart Diagram

5) Implementation

Screenshot of Working Project

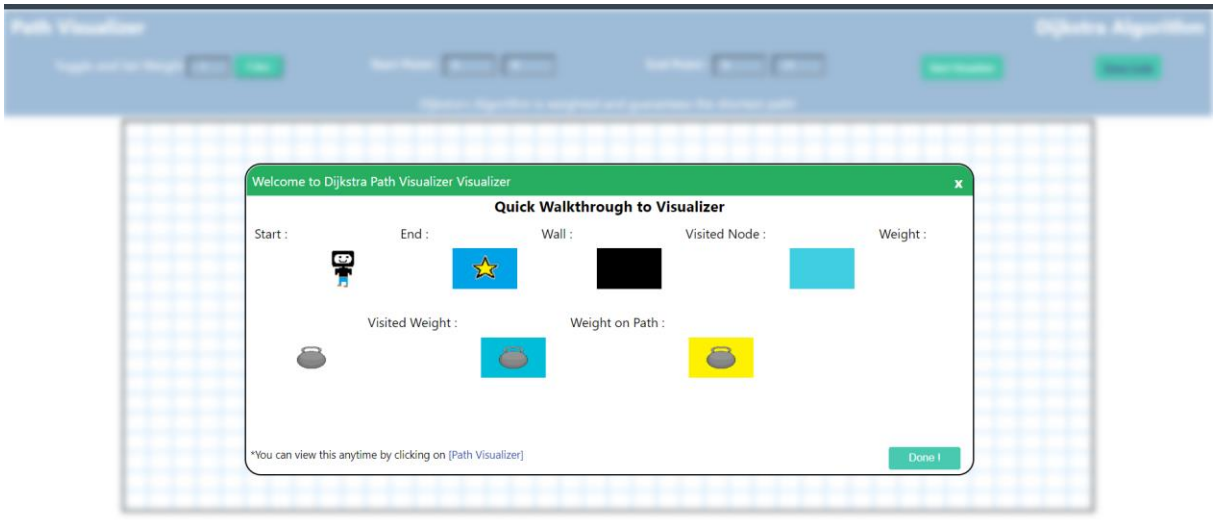

Fig17.1 Tutorial Screen

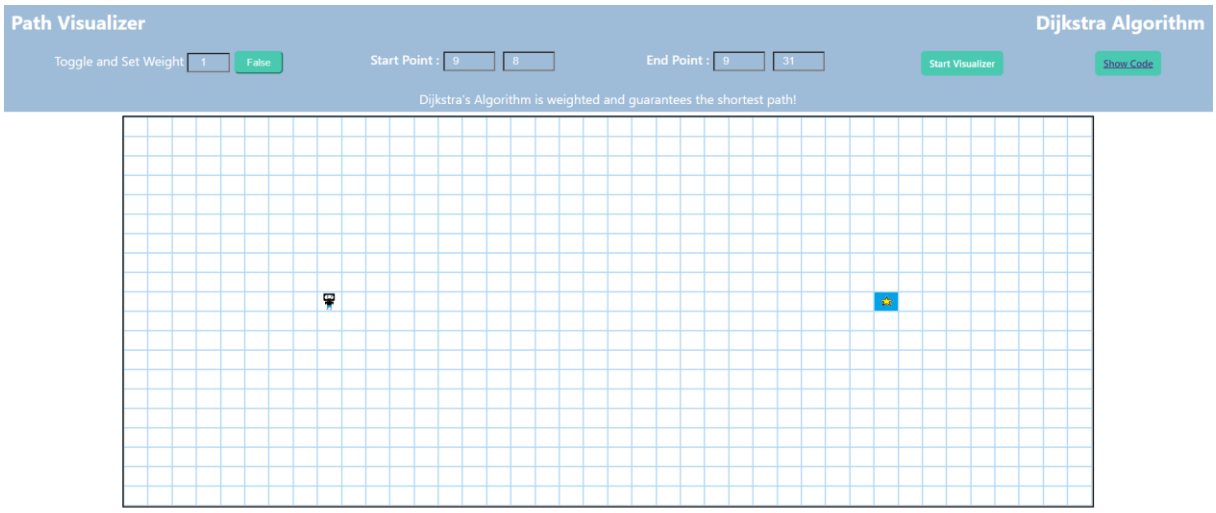

Fig 17.2 Grid Screen

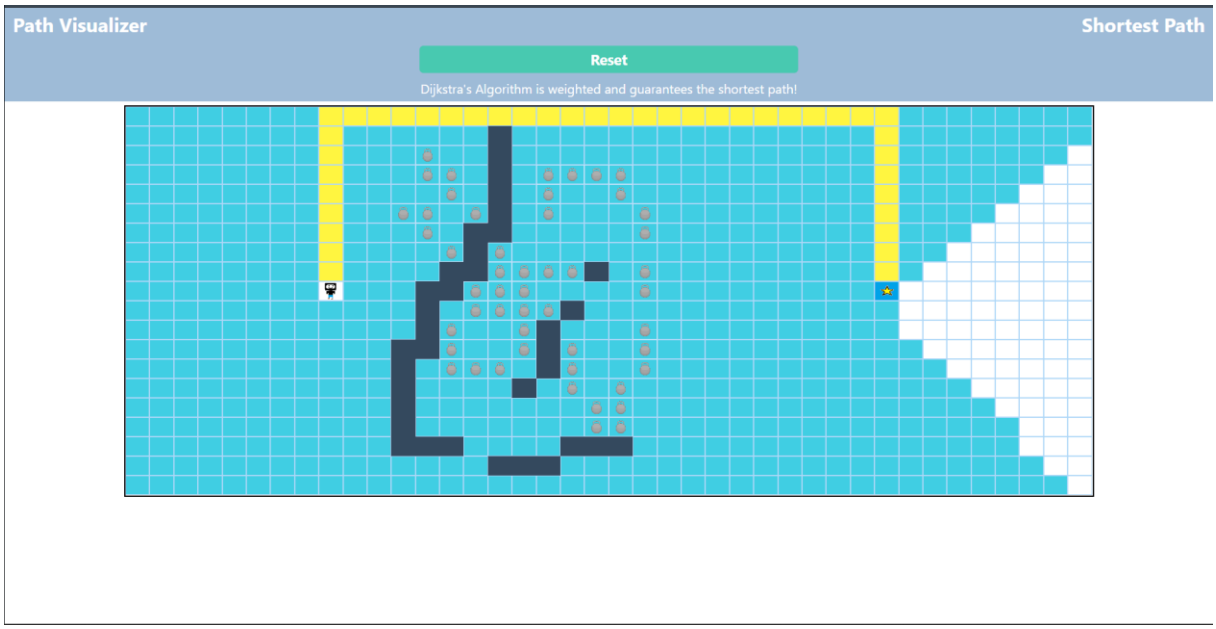

Fig 17.3 Visualizer Screen

6) Testing

Testing

Test Case #: 1	Test Case Name: Tutorial
System: Tutorial Screen	Subsystem: Tutorial
Designed by: Sartaj	Design Date: 3rd October 2022
Executed by: Piyush	Execution Date: 6th October 2022
Short Description: Tutorial Screen for new users.	

Pre-conditions:
The website should be open.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Clicking on 'Done'	The tutorial screen should be closed	**Pass**	
2	Refresh Page	The tutorial screen should display	**Pass**	

Post-conditions
The website is now ready for setting the Start and Goal node.
The website is now ready for visualizing the path.

Test Case #: 2

Test Case Name: Show Code

System: Show Code

Subsystem: Show Code

Designed by: Maninder

Design Date: 3rd November 2022

Executed by: Piyush

Execution Date: 6th October 2022

Short Description: Showing code for users

Pre-conditions:
The website should be open.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Clicking on 'Show Code'	The screen should prompts an another tab for showing code.	**Fail**	

Post-conditions
The website is now ready for setting the Start and Goal node.
The website is now ready for visualizing the path.

Test Case #: 3

Test Case Name: Add resistance

System: Add Resistance

Subsystem: Add Weight

Designed by: Sukhvir Singh

Design Date: 23rd August 2022

Executed by: Piyush

Execution Date: 6th October 2022

Short Description: Add weight to grid.

Pre-conditions:
1) The Start and goal node should be set.

Step	Action	Expected System Response	Pass/ Fail	Comment
1	Clicking on 'True' button	The system should true the weight value.	**Pass**	
2	Clicking on "False" Button	The system should false the weight value.	**Pass**	
3	Clicking on grid	The system should add the weight on the grid	**Pass**	

Post-conditions
The website is now ready for setting the Start and Goal node.
The website is now ready for visualizing the path.