```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.SceneManagement;
public class UIManager : MonoBehaviour
{
    [SerializeField] private GameObject backGround;
    [SerializeField] private Text distance;

    [SerializeField] private Dropdown startDropDown;
    [SerializeField] private Dropdown destinationDropDown;
    private Graph graph;

    private Node startingNode;

    // a toggle to displaying GUI elements
    private bool isGUIVisible;


    void Start() {
        backGround.SetActive(false);
        graphInit();
        addVertex();
        addEdge();
        isGUIVisible = true;
        addOption();
    }

    public void addOption() {
        // add all of the vertices to start drop down
        foreach(Node room in graph.getVertices()) {
            startDropDown.options.Add(new Dropdown.OptionData()
{text=room.getData()});
            // also add all the room to destinationDropDown
            destinationDropDown.options.Add(new Dropdown.OptionData()
{text=room.getData()});
        }
    }

    public void graphInit() {
        graph = new Graph();

    }

    public void quitGame() {
        Application.Quit();
    }

    public void addEdge() {
        graph.addEdge(graph.getNodeByValue("Main Entrance"),
graph.getNodeByValue("Reception"), 3);
        graph.addEdge(graph.getNodeByValue("Reception"),
graph.getNodeByValue("Office"), 2);
        graph.addEdge(graph.getNodeByValue("Office"),
graph.getNodeByValue("Toilet"), 2);
        graph.addEdge(graph.getNodeByValue("Main Entrance"),
graph.getNodeByValue("Exit"), 5);
        graph.addEdge(graph.getNodeByValue("Reception"),
graph.getNodeByValue("Exit"), 3);
```

```
54          graph.addEdge(graph.getNodeByValue("Office"),
graph.getNodeByValue("Exit"), 2);
55          graph.addEdge(graph.getNodeByValue("Toilet"),
graph.getNodeByValue("Exit"), 3);
56          graph.addEdge(graph.getNodeByValue("Exit"), graph.getNodeByValue("Lift0"),
2);
57          graph.addEdge(graph.getNodeByValue("Main Exit"),
graph.getNodeByValue("Lift0"), 5);
58          graph.addEdge(graph.getNodeByValue("Lift1"),
graph.getNodeByValue("Lift0"), 6);
59          graph.addEdge(graph.getNodeByValue("Front Desk"),
graph.getNodeByValue("Lift0"), 4);
60          graph.addEdge(graph.getNodeByValue("Front Desk"),
graph.getNodeByValue("Main Exit"), 8);
61          graph.addEdge(graph.getNodeByValue("Lift1"), graph.getNodeByValue("Lecture
Room 1"), 4);
62          graph.addEdge(graph.getNodeByValue("Lift1"), graph.getNodeByValue("Lecture
Room 2"), 8);
63          graph.addEdge(graph.getNodeByValue("Lift1"), graph.getNodeByValue("Lecture
Room 3"), 4);
64          graph.addEdge(graph.getNodeByValue("Lecture Room 3"),
graph.getNodeByValue("Lecture Room 1"), 3);
65          graph.addEdge(graph.getNodeByValue("Lecture Room 3"),
graph.getNodeByValue("Lecture Room 2"), 3);
66      }
67
68
69      public void addVertex() {
70          // add all the objects of given name to the collection
71          graph.addVertex("Main Entrance");
72          graph.addVertex("Reception");
73          graph.addVertex("Office");
74          graph.addVertex("Toilet");
75          graph.addVertex("Lift0");
76          graph.addVertex("Lift1");
77          graph.addVertex("Main Exit");
78          graph.addVertex("Exit");
79          graph.addVertex("Front Desk");
80          graph.addVertex("Lecture Room 1");
81          graph.addVertex("Lecture Room 2");
82          graph.addVertex("Lecture Room 3");
83      }
84
85
86      public void OnClickPlay() {
87          isGUIVisible = false;
88          backGround.SetActive(true);
89      }
90
91      public void OnClickTutorial() {
92           SceneManager.LoadScene(sceneName:"Tutorial Scene");
93      }
94
95
96
97
98      public void OnSubmitStart(string startNode) {
99          // this function remembers where user is and save in the UIManger script
100         startingNode = graph.getNodeByValue(startNode);
101     }
```

```
102
103     public void OnSubmitDestination(string destinationNode) {
104        isGUIVisible = true;
105        StartCoroutine(displayColour(destinationNode));
106        StartCoroutine(displayHalo(destinationNode));
107     }
108
109     public IEnumerator displayHalo(string destinationNode) {
110         Node destination = graph.getNodeByValue(destinationNode);
111         // calculate the shortest path and save in a variable
112         List<Node> path = Algorithm.findShortestPath(graph, startingNode,
destination);
113         foreach(Node v in path) {
114             GameObject obj = GameObject.Find(v.getData());
115             Behaviour halo = (Behaviour)obj.GetComponent("Halo");
116             halo.enabled = true;
117             yield return new WaitForSeconds(1f);
118             halo.enabled = false;
119         }
120     }
121
122     /*
123     Auxiliary function from https://answers.unity.com/questions/8338/how-to-draw-
a-line-using-script.html
124     that helps to draw a line segment between two points
125     */
126     public void DrawLine(Vector3 start, Vector3 end, Color color)
127          {
128              GameObject myLine = new GameObject();
129              myLine.transform.position = start;
130              myLine.AddComponent<LineRenderer>();
131              LineRenderer lr = myLine.GetComponent<LineRenderer>();
132              lr.material = new Material(Shader.Find("Sprites/Default"));
133              lr.startColor = color;
134              lr.endColor =color;
135              lr.startWidth = 0.5f;
136              lr.endWidth = 0.5f;
137              lr.SetPosition(0, start);
138              lr.SetPosition(1, end);
139              GameObject.Destroy(myLine,5f);
140          }
141
142
143     public IEnumerator displayColour(string destinationNode) {
144         // set the user-prompt screen off
145         backGround.SetActive(false);
146         // switch on the green colour of the starting node
147         GameObject start = GameObject.Find(startingNode.getData());
148         start.GetComponent<Renderer>().material = new
Material(Shader.Find("Sprites/Default"));
149         start.GetComponent<Renderer>().material.SetColor("_Color", Color.green);
150         // switch on the destinationNode colour to red
151         Node destination = graph.getNodeByValue(destinationNode);
152         GameObject finish = GameObject.Find(destination.getData());
153         finish.GetComponent<Renderer>().material = new
Material(Shader.Find("Sprites/Default"));
154         finish.GetComponent<Renderer>().material.SetColor("_Color", Color.red);
155         // calculate the shortest path and save in a variable
156         List<Node> path = Algorithm.findShortestPath(graph, startingNode,
destination);
```

```
157         // once we have the path, we will make the Node glow in a order to
illustrate the direction should be taken
158         // first, we need reference to each object, which we already have
159         // iterate through the path, turn on the halo property, wait for 1.5
seconds, then turn it off again
160         for(int i=0; i<path.Count-1; i++) {
161             GameObject obj = GameObject.Find(path[i].getData());
162             // draw a line between all vertices in the path
163             DrawLine(obj.transform.position,
GameObject.Find(path[i+1].getData()).transform.position, Color.blue);
164         }
165         // wait for 5 seconds and reset the colors
166         yield return new WaitForSeconds(5f);
167         start.GetComponent<Renderer>().material.SetColor("_Color", Color.black);
168         finish.GetComponent<Renderer>().material.SetColor("_Color", Color.black);
169         // once the shortest path is calculated, distance should also be update on
the left corner
170         distance.text = "The Shortest distance to " + destinationNode + " is " +
Algorithm.dijkstra(graph,
startingNode).shortestDistanceEstimate[destination].ToString() + " meters";
171     }
172
173        // label each object in the list
174         void OnGUI() {
175             if(isGUIVisible) {
176                 List<Node> vertices = graph.getVertices();
177                 foreach(Node v in vertices) {
178                     var style = new GUIStyle();
179                     style.fontSize = 50;
180                     GameObject obj = GameObject.Find(v.getData());
181                     Rect display = new Rect(0,0,200,100);
182                     Vector3 location =
Camera.main.WorldToScreenPoint(obj.transform.position + new Vector3(0,0,0.5f));
183                     display.x = location.x;
184                     display.y = Screen.height - location.y - display.height;
185                     //GUIStyle border = new GUIStyle(GUI.skin.label);
186                     //border.margin = new RectOffset(10,10,10,10);
187                     GUI.Label(display, v.getData(), style);
188                 }
189             }
190         }
191
192 }
193
```