Assignment Report

- **System Design And Architectural Details:**
  - **Coordinate system:** Since the left top-most corner is by default origin in the pygame window therefore, the below coordinate system is chosen for convenience.

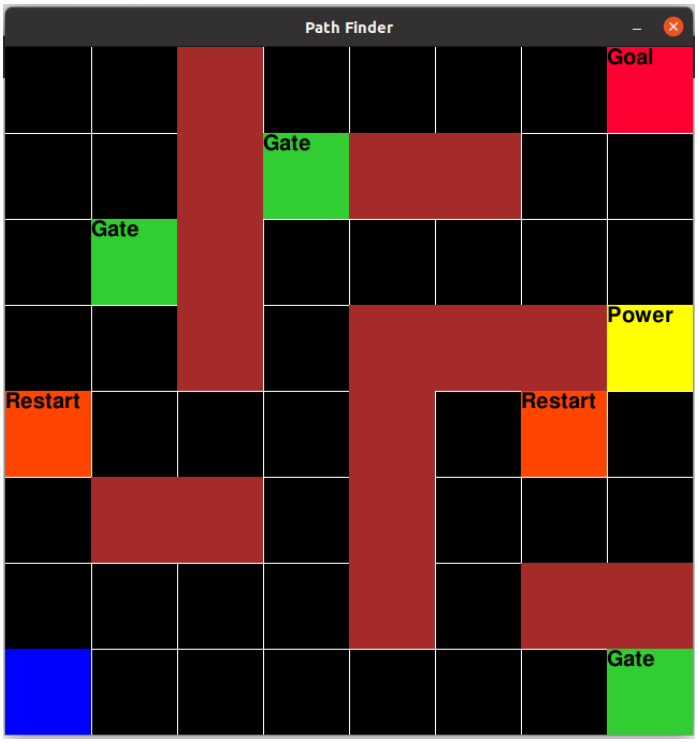
(1,1)	(1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(1,7)	(1,8)
(2,1)	(2,2)	(2,3)	(2,4)	(2,5)	(2,6)	(2,7)	(2,8)
(3,1)	(3,2)	(3,3)	(3,4)	(3,5)	(3,6)	(3,7)	(3,8)
(4,1)	(4,2)	(4,3)	(4,4)	(4,5)	(4,6)	(4,7)	(4,8)
(5,1)	(5,2)	(5,3)	(5,4)	(5,5)	(5,6)	(5,7)	(5,8)
(6,1)	(6,2)	(6,3)	(6,4)	(6,5)	(6,6)	(6,7)	(6,8)
(7,1)	(7,2)	(7,3)	(7,4)	(7,5)	(7,6)	(7,7)	(7,8)
(8,1)	(8,2)	(8,3)	(8,4)	(8,5)	(8,6)	(8,7)	(8,8)

  - **Color Code Representation:** The boxes with different colors represent different blocks which is described in below table and an example is also shown:

Color	Representation
Blue	Agent Location
Yellow	Power Position
Orange	Restart

Black	Empty Position
Red	Goal Position
Green	Power Position Gate
Brown	Wall

For example:



- **<u>Policy Design</u>:** For learning the optimal policy using reinforcement learning, TD Q-learning is applied. The equation for TD Q-learning is as follows:

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\left(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} \right)}^{\text{new value (temporal difference target)}} - \underbrace{Q(s_t, a_t)}_{\text{old value}}}^{\text{temporal difference}}$$

The results obtained by the Q-learning are stored in a Q-table which later is used to find the optimal action to be taken at the current position.

- **Knowledge Base:** There is no separate variable used to account for the knowledge base captured but Q-table itself acts as the knowledge base with a highly negative value for an action resulting in colliding with the wall or stepping on restart position etc while containing a highly positive value for an action resulting in reaching the goal.

- **Reward Function:** The reward function is designed to give an immediate reward to the agent depending on the consequences of the action taken on the current state. The reward given v/s the consequences of the action is as follows:

Action Effect	Reward
Collision with wall	-50
Collision with boundaries	-50
Stepping on restart	-50
Reaching on goal	100
Stepping on powergate	Reward for power position.
Stepping on normal location	Negative of absolute manhattan distance of position from goal.

Here the negative reward given to the agent for a normal move also ensures that the agent takes the shortest path.
The python code for the reward function is as follows:

```
def getReward(pos, action, wall, restart, powergate, goal,
powerpos):
      x = pos[0]
      y = pos[1]
      #up = 1
      #down = 2
      #right = 3
      #left = 4
      if action == 0:
```

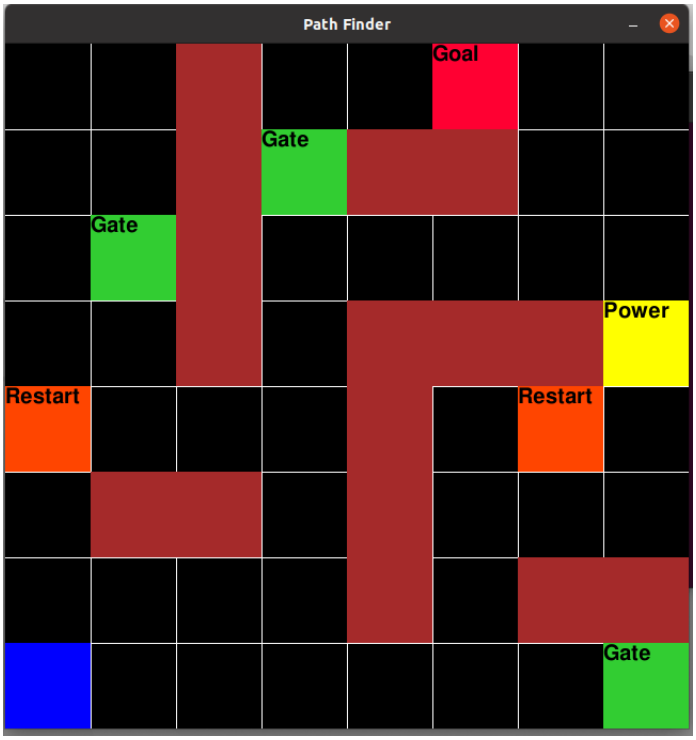
```
            y = y-1
      elif action == 1:
            y = y+1
      elif action == 2:
            x = x+1
      elif action == 3:
            x = x-1

      if x<1 or x>8:
            return -50
      if y<1 or y>8:
            return -50
      if (x,y) in wall:
            return -50
      if (x,y) in restart:
            return -50
      if (x,y) in goal:
            return 100
      if (x,y) in powergate:
            return
(abs(goal[0][0]-powerpos[0][0])+abs(goal[0][1]-powerpos[0][1])
)*(-1)
      return (abs(goal[0][0]-x)+abs(goal[0][1]-y))*(-1)
```

- **Hyper-parameters:** The agent's learning depends on the choice of many hyper parameters which are as follows:

Hyper Parameter	Value
Learning Rate	0.9
Discount Factor	0.8

● Observations:

- The observations for three iterations i.e independent runs are as follows:
  - **First Iteration:**
    - **Initial State:**



- **Q-table:**

The learned q-table is:
x = 1 y= 1 action= 0 value= -81.0
x = 1 y= 1 action= 1 value= -5.4
x = 1 y= 1 action= 2 value= -3.6
x = 1 y= 1 action= 3 value= -81.0
x = 1 y= 2 action= 0 value= -7.0920000000000005
x = 1 y= 2 action= 1 value= -6.3
x = 1 y= 2 action= 2 value= -4.5
x = 1 y= 2 action= 3 value= -81.0
x = 1 y= 3 action= 0 value= -8.64
x = 1 y= 3 action= 1 value= -7.2
x = 1 y= 3 action= 2 value= -4.5
x = 1 y= 3 action= 3 value= -81.0
x = 1 y= 4 action= 0 value= -9.54
x = 1 y= 4 action= 1 value= -45.0
x = 1 y= 4 action= 2 value= -6.3
x = 1 y= 4 action= 3 value= -81.0
x = 1 y= 6 action= 0 value= -45.0
x = 1 y= 6 action= 1 value= -9.9
x = 1 y= 6 action= 2 value= -45.0
x = 1 y= 6 action= 3 value= -81.0
x = 1 y= 7 action= 0 value= -16.128000000000004
x = 1 y= 7 action= 1 value= -10.8
x = 1 y= 7 action= 2 value= -9.0
x = 1 y= 7 action= 3 value= -81.0
x = 1 y= 8 action= 0 value= -16.38

x = 1 y= 8 action= 1 value= -81.0
x = 1 y= 8 action= 2 value= -9.9
x = 1 y= 8 action= 3 value= -81.0
x = 2 y= 1 action= 0 value= -81.0
x = 2 y= 1 action= 1 value= -4.5
x = 2 y= 1 action= 2 value= -45.0
x = 2 y= 1 action= 3 value= -7.0920000000000005
x = 2 y= 2 action= 0 value= -6.84
x = 2 y= 2 action= 1 value= -4.5
x = 2 y= 2 action= 2 value= -45.0
x = 2 y= 2 action= 3 value= -8.64
x = 2 y= 4 action= 0 value= -4.5
x = 2 y= 4 action= 1 value= -7.2
x = 2 y= 4 action= 2 value= -45.0
x = 2 y= 4 action= 3 value= -11.735999999999999
x = 2 y= 5 action= 0 value= -9.54
x = 2 y= 5 action= 1 value= -45.0
x = 2 y= 5 action= 2 value= -6.3
x = 2 y= 5 action= 3 value= -45.0
x = 2 y= 7 action= 0 value= -45.0
x = 2 y= 7 action= 1 value= -9.9
x = 2 y= 7 action= 2 value= -8.1
x = 2 y= 7 action= 3 value= -16.38
x = 2 y= 8 action= 0 value= -14.832
x = 2 y= 8 action= 1 value= -81.0
x = 2 y= 8 action= 2 value= -9.0
x = 2 y= 8 action= 3 value= -17.928
x = 3 y= 5 action= 0 value= -45.0
x = 3 y= 5 action= 1 value= -45.0
x = 3 y= 5 action= 2 value= -5.4
x = 3 y= 5 action= 3 value= -11.735999999999999
x = 3 y= 7 action= 0 value= -45.0
x = 3 y= 7 action= 1 value= -9.0
x = 3 y= 7 action= 2 value= -7.2
x = 3 y= 7 action= 3 value= -14.832
x = 3 y= 8 action= 0 value= -13.284000000000002
x = 3 y= 8 action= 1 value= -81.0
x = 3 y= 8 action= 2 value= -8.1
x = 3 y= 8 action= 3 value= -16.38
x = 4 y= 1 action= 0 value= -81.0
x = 4 y= 1 action= 1 value= -4.5
x = 4 y= 1 action= 2 value= -0.9
x = 4 y= 1 action= 3 value= -45.0
x = 4 y= 3 action= 0 value= -4.5
x = 4 y= 3 action= 1 value= -4.5
x = 4 y= 3 action= 2 value= -2.7
x = 4 y= 3 action= 3 value= -45.0
x = 4 y= 4 action= 0 value= -5.5440000000000005
x = 4 y= 4 action= 1 value= -5.4
x = 4 y= 4 action= 2 value= -45.0
x = 4 y= 4 action= 3 value= -45.0
x = 4 y= 5 action= 0 value= -8.388
x = 4 y= 5 action= 1 value= -6.3
x = 4 y= 5 action= 2 value= -45.0

x = 4 y= 5 action= 3 value= -10.188
x = 4 y= 6 action= 0 value= -9.936
x = 4 y= 6 action= 1 value= -7.2
x = 4 y= 6 action= 2 value= -45.0
x = 4 y= 6 action= 3 value= -45.0
x = 4 y= 7 action= 0 value= -11.484000000000002
x = 4 y= 7 action= 1 value= -8.1
x = 4 y= 7 action= 2 value= -45.0
x = 4 y= 7 action= 3 value= -13.284000000000002
x = 4 y= 8 action= 0 value= -13.032
x = 4 y= 8 action= 1 value= -81.0
x = 4 y= 8 action= 2 value= -7.2
x = 4 y= 8 action= 3 value= -14.832
x = 5 y= 1 action= 0 value= -81.0
x = 5 y= 1 action= 1 value= -45.0
x = 5 y= 1 action= 2 value= 90.0
x = 5 y= 1 action= 3 value= -2.448000000000000
x = 5 y= 3 action= 0 value= -45.0
x = 5 y= 3 action= 1 value= -45.0
x = 5 y= 3 action= 2 value= -1.8
x = 5 y= 3 action= 3 value= -5.5440000000000005
x = 5 y= 8 action= 0 value= -45.0
x = 5 y= 8 action= 1 value= -81.0
x = 5 y= 8 action= 2 value= -6.3
x = 5 y= 8 action= 3 value= -13.284000000000002
x = 6 y= 3 action= 0 value= -45.0
x = 6 y= 3 action= 1 value= -45.0
x = 6 y= 3 action= 2 value= -2.7
x = 6 y= 3 action= 3 value= -3.9960000000000004
x = 6 y= 5 action= 0 value= -45.0
x = 6 y= 5 action= 1 value= -4.5
x = 6 y= 5 action= 2 value= -45.0
x = 6 y= 5 action= 3 value= -45.0
x = 6 y= 6 action= 0 value= -6.84
x = 6 y= 6 action= 1 value= -5.4
x = 6 y= 6 action= 2 value= -5.4
x = 6 y= 6 action= 3 value= -45.0
x = 6 y= 7 action= 0 value= -8.388
x = 6 y= 7 action= 1 value= -6.3
x = 6 y= 7 action= 2 value= -45.0
x = 6 y= 7 action= 3 value= -45.0
x = 6 y= 8 action= 0 value= -9.936
x = 6 y= 8 action= 1 value= -81.0
x = 6 y= 8 action= 2 value= -7.2
x = 6 y= 8 action= 3 value= -11.735999999999999
x = 7 y= 1 action= 0 value= -81.0
x = 7 y= 1 action= 1 value= -1.8
x = 7 y= 1 action= 2 value= -1.8
x = 7 y= 1 action= 3 value= 90.0
x = 7 y= 2 action= 0 value= 63.9
x = 7 y= 2 action= 1 value= -2.7
x = 7 y= 2 action= 2 value= -2.7
x = 7 y= 2 action= 3 value= -45.0
x = 7 y= 3 action= 0 value= 44.208000000000006

x = 7 y= 3 action= 1 value= -45.0
x = 7 y= 3 action= 2 value= -3.6
x = 7 y= 3 action= 3 value= -3.744
x = 7 y= 6 action= 0 value= -45.0
x = 7 y= 6 action= 1 value= -45.0
x = 7 y= 6 action= 2 value= -6.3
x = 7 y= 6 action= 3 value= -8.388
x = 7 y= 8 action= 0 value= -45.0
x = 7 y= 8 action= 1 value= -81.0
x = 7 y= 8 action= 2 value= -4.5
x = 7 y= 8 action= 3 value= -11.484000000000002
x = 8 y= 1 action= 0 value= -81.0
x = 8 y= 1 action= 1 value= -2.7
x = 8 y= 1 action= 2 value= -81.0
x = 8 y= 1 action= 3 value= 63.9
x = 8 y= 2 action= 0 value= 44.208000000000006
x = 8 y= 2 action= 1 value= -3.6
x = 8 y= 2 action= 2 value= -81.0
x = 8 y= 2 action= 3 value= 44.208000000000006
x = 8 y= 3 action= 0 value= 29.129760000000005
x = 8 y= 3 action= 1 value= -4.5
x = 8 y= 3 action= 2 value= -81.0
x = 8 y= 3 action= 3 value= 29.129760000000005
x = 8 y= 4 action= 0 value= 17.373427200000005
x = 8 y= 4 action= 1 value= -5.4
x = 8 y= 4 action= 2 value= -81.0
x = 8 y= 4 action= 3 value= -45.0
x = 8 y= 5 action= 0 value= 8.008867584000004
x = 8 y= 5 action= 1 value= -6.3
x = 8 y= 5 action= 2 value= -81.0
x = 8 y= 5 action= 3 value= -45.0
x = 8 y= 6 action= 0 value= 0.3663846604800033
x = 8 y= 6 action= 1 value= -45.0
x = 8 y= 6 action= 2 value= -81.0
x = 8 y= 6 action= 3 value= -9.936

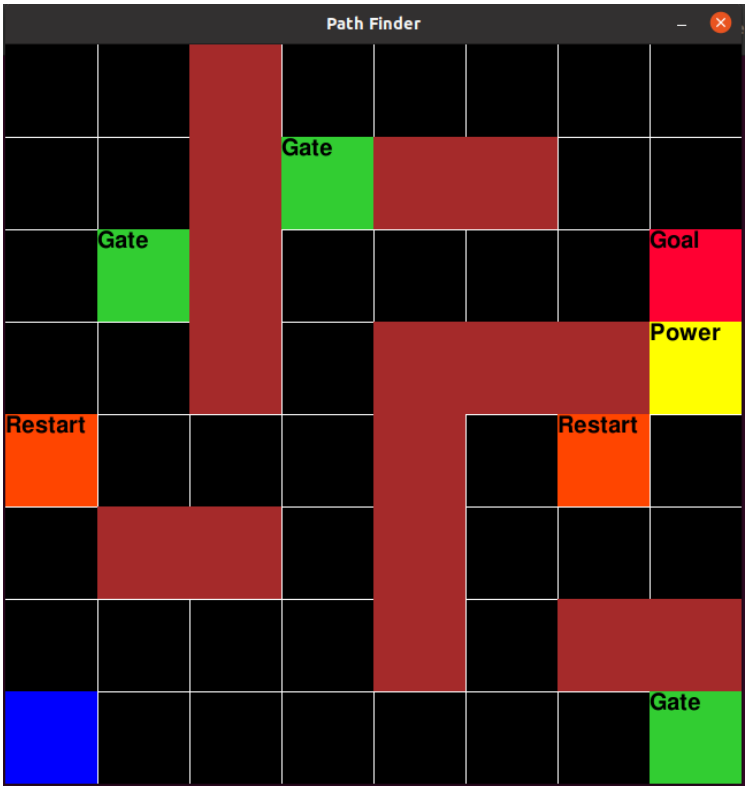
Where action 0 : “UP ” , action 1 : “Down ” , action 2 : “Right ” , action 3 : “Left ”

- **<u>Path Followed</u>:**

  Start (1,8) → (2,8) → (3,8) → (4,8) → (5,8) → (6,8) → (7,8) → (8,4) → (8,3) → (8,2) → (8,1) → (7,1) Goal

- **<u>Path Cost</u>:** 192.41

- **Second Iteration:**
  - **<u>Initial State</u>:**

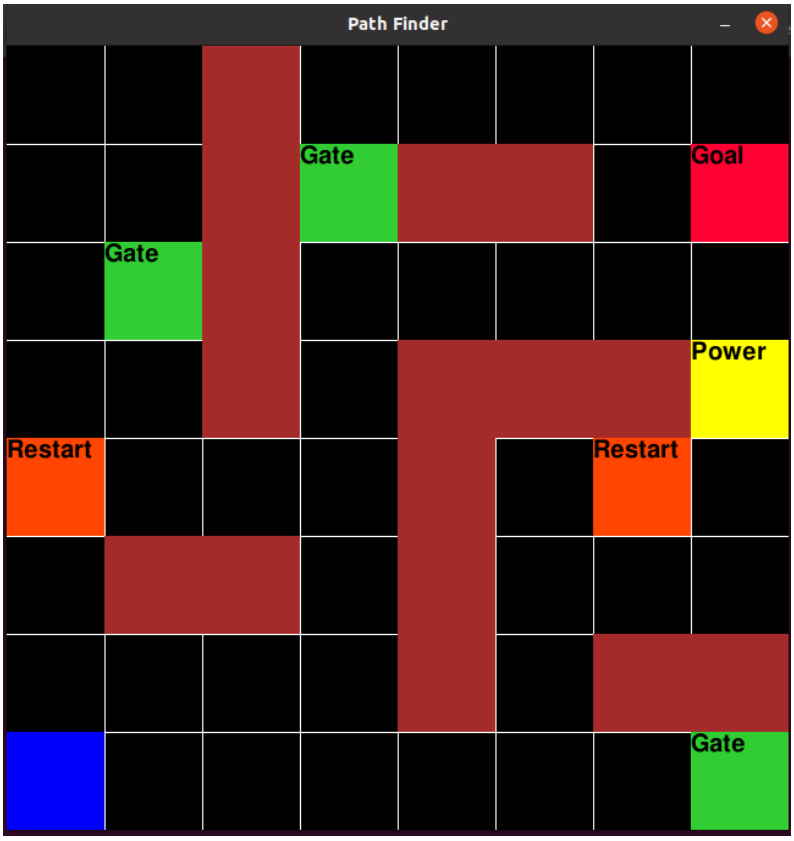


- **<u>Path Followed</u>:**

  Start (1,8) → (2,8) → (3,8) → (4,8) → (5,8) → (6,8) → (7,8) → (8,4) Goal

- **<u>Path Cost</u>:** 43.2

- **Third Iteration:**
  - **<u>Initial State</u>:**



- **<u>Path Followed</u>:**

  Start (1,8) → (2,8) → (3,8) → (4,8) → (5,8) → (6,8) → (7,8) → (8,4) → (8,3)Goal

- **<u>Path Cost</u>:** 100.8