DESIGN AND IMPLEMENTATION OF VEHICLE PARKING SYSTEM

Submitted in partial fulfilment of the requirements for the degree of

Master of technology

In

M-Tech integrated software engineering

By

NAME	REGISTRATION NO
16MIS0133	M SAI DEEPU
16MIS0256	M NIKHIL CHAKRAVARTHY

Under the guidance of

Prof. GUNASEKARAN G

SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING

VIT, VELLORE

APRIL 2019

DECLARATION

I hereby declare that the thesis entitled "DESIGN AND IMPLEMENTATION OF VEHICLE PARKING SYSTEM" submitted by me, for the award of the degree of *Master of Technology integrated software engineering* to VIT is a record of bonafide work carried out by me under the supervision of PROF. GUNASEKARAN G.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date :

Signature of the Candidates

CERTIFICATE

This is to certify that the thesis entitled "DESIGN AND IMPLEMENTATION OF VEHICLE PARKING SYSTEM" submitted by **M SAI DEEPU (16MIS0133), M NIKHIL CHAKRAVARTHY (16MIS0256) , SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**, VIT University, for the award of the degree of *Master of Technology in Integrated software engineering*, is a record of bonafide work carried out by him under my supervision during the period, 01. 12. 2018 to 30.04.2019, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place: Vellore

Date:

Signature of the Guide

CONTENTS **Page No**

1. INTRODUCTION:

1.1 OBJECTIVE:

In 1997, the Department of the Environment, Transport and the Regions (DETR), together with the Traffic Director for London, launched the Urban Traffic Management and Control (UTMC) initiative. This Policy-sensitive traffic signal control project, which began in March 1997 and was conducted in the context of proper traffic management. The project's overall objective was to develop a traffic signal controller, which can be sensitive to changing traffic management policy, and aim to achieve a wide range of transport and environmental objectives in addition to minimisation of vehicle delays and stops. Its specific objectives were to:

(1) apply the simulation to seek expert consensus on the objectives of traffic signal control;

(2) develop a prototype fuzzy logic traffic signal controller with four specified features;

(3) optimise the controller's membership functions and/or rule bases, using a simulated road network in Newcastle; and

(4) evaluate the controller's performance by simulation and test it against SCOOT.

Our project concentrates on the simulation of the traffic system, which helps in reducing the proper management of the traffic, which used two questionnaires; although this study found no expert consensus, it emphasised that flexibility is one of the most important features of a traffic signal control strategy.

It was basically the integration of the traffic signal system with this parking entry system. Smart vehicle parking management systems are capable of providing extreme level of convenience to the drivers. In this project the parking system is integrated with traffic signals. As in any traffic signal the three lights - red, yellow and green. All the options have been implemented. To regulate the traffic of vehicle entering into Parking cellular. Both mouse and keyboard interaction has been added to the OpenGL Code. Using the interactions between the keyboard and mouse the flow of traffic were kept under control in the simulation. This helps in better understanding of the traffic signal system.

1.2 MOTIVATION

In today’s technological world the concept of smart city has become an area of interest. Concern to parking became impending in an urban area. The parking space problem can be turn into a new opportunity brought by the recent trends to meet the world’s connected

continuum. Traditional parking system commonly uses security ultrasonic sensors, camera or infrared ray sensors to manage the parking areas.

1.3 BACKGROUND

Automated Parking slot helps in the regulation of any parking slot with the large number of vehicle involved. This project concentrates on less manpower utilisation, less installation and maintenance cost. This project is a versatile project which can be implemented on any and location. This is a conceptual project and just a prototype was designed and hence can go through the various stages of modification to make it reliable and much more suitable for the real time applications [1].

An automatic parking system based on parking scene recognition is proposed in this paper to resolve the following issues with existing automatic parking systems: parking scene recognition methods are less intelligent, vehicle control has a low degree of automation, and the research scope is limited to traditional fuel vehicles. To increase the utilization of parking spaces and parking convenience, machine vision and pattern recognition techniques are introduced to intelligently recognize a vertical parking scenario, plan a reasonable parking path, develop a path tracking control strategy to improve the vehicle control automation, and explore a highly intelligent automatic parking technology roadmap. This paper gives three key aspects of system solutions for an automatic parking system based on parking scene recognition: parking scene recognition, parking path planning, and path tracking and control [2].

This paper has shown the concept of an automatic car parking system. Everything in the modern world is going automatic, we have built a system which can automatically sense the entry and exit of cars through the gate and then display the of cars in the parking lot. This automated car parking system reduces the time taken to check the space for vehicles by displaying the available spaces for parking on a LCD displayer by using infrared (IR) sensors installed at the entrance and exit. This project is developed using 89c52 microcontroller [3].

In this paper, the author proposed web App system, named “Park Easy” is based on the usage of smart phones, sensors monitoring techniques with a camera which is used as a sensor to take photos to show the occupancy of cars parks. By implementing this system, the utilization of parking spaces will increase. It allocates available parking space to a given driver to park their vehicle, renew the availability of the parking space when the car leaves and compute the charges due.Smart parking App, “Park Easy”, will also enable most important techniques to

provide all the possible shortage route for parking from any area of the city mainly, it helps to predict accurately and sense spot/vehicle occupancy in real-time [4].

In this paper the author presented the IoT based Smart Car Parking System. This paper makes easy for the user to find automatically a free space at the low cost and without consuming time and fuel. The whole system is based on Wi-Fi network. The android application in mobile is also provided to the user to check the availability of free space for parking and book that slot accordingly [5].

2. PROJECT DESCRIPTION AND GOALS

In this project they implemented traffic signal concept to smart parking system to reduce the traffic of vehicles entering into the parking cellular. This system gives clear indication to drivers when to enter into parking lane, when to stop. In this project the parking system is integrated with traffic signals. As in any traffic signal the three lights - red, yellow and green. All the options have been implemented. To regulate the traffic of vehicle entering into Parking cellular. Both mouse and keyboard interaction has been added to the OpenGL Code.

Goals of the project is to give clear cut information to the driver when the vehicle to be parked in the cellular. If suppose already a vehicles entered in the cellular. Our system will signal the next vehicle to stop for a while until other vehicle is parked. So this system will be helpful to the vehicle drivers to keep vehicle safe without dashing to the other vehicles due to miss understandings or traffic over there.

3. TECHNICAL SPECIFICATION

WINDOWS 10,

8 GB RAM,

500 GB ROM,

OPENGL,

CODE BLOCKS OR OTHER C++ CODE COMPILERS

4. DESIGN APPROACH AND DETAILS

4.1 Design Approach

In any traffic signal the three lights - red, yellow and green. All the options has been implemented. To regulate the traffic of vehicle entering into Parking cellular. Both mouse and keyboard interaction has been added to the OpenGL Code.

User Interactions

As mentioned earlier both mouse and keyboard user interaction has been added to this OpenGL Code.

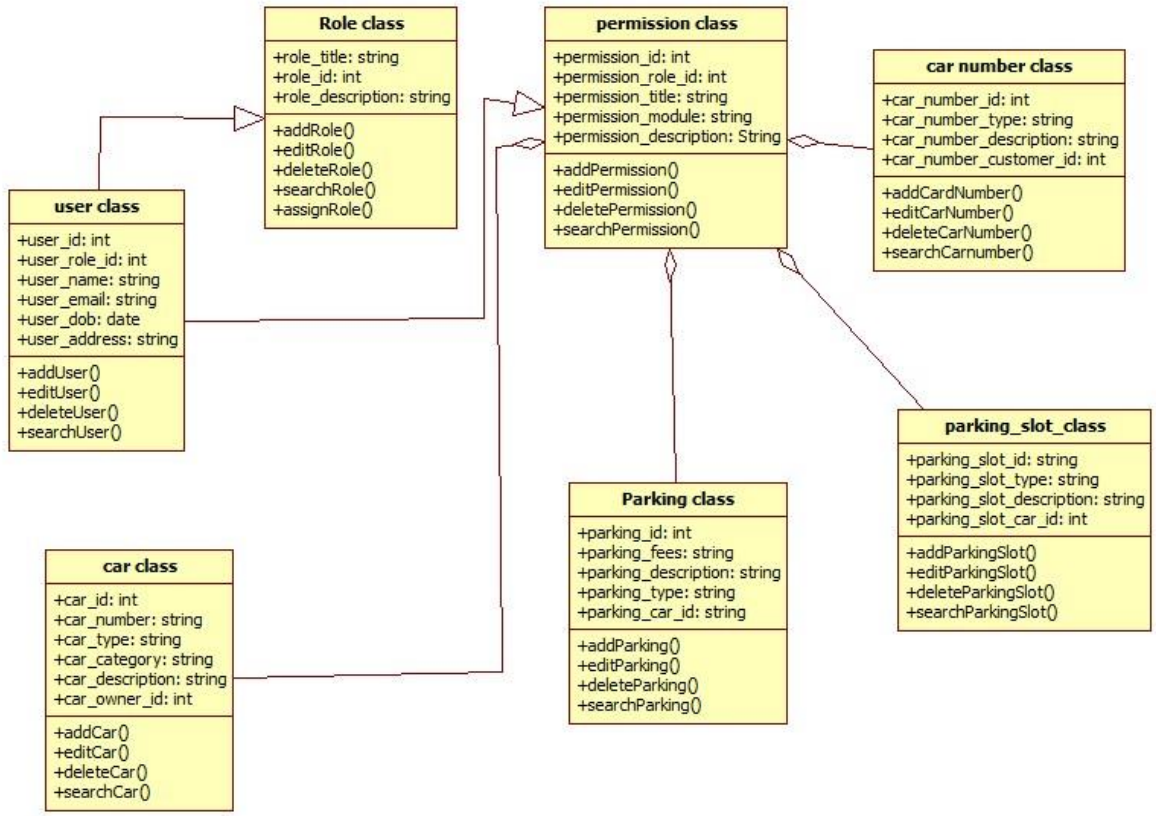
a) Keyboard Interactions

- **Enter -** From First Introduction screen to screen press Enter key at beginning.
- **Help -** Press **'h'** to get the help screen.
- **Left to right movement -** press **'l'** to allow only left to right movement of traffic.
- **Right to left movement -** press **'r'** to allow only right to left movement of traffic.
- **Speed up -** To speed up the traffic press **'s'**.

b) Mouse Interactions

- **Left Mouse Button -** This will stop the traffic as Red light gets on.
- **Right Mouse Button (on hold) -** Press right mouse button and keep on hold for yellow light.
- **Right Mouse Button (released) -** After releasing the right mouse button the light changes from yellow to green and traffic moves.

Class Diagram



4.2 CODE AND STANDARDS
SOURCE CODE

```
#include<windows.h>
#include<stdio.h>
#include<GL/glut.h>



void bus();
void road();
void signal();
void car();
void car2();
void mydisplay();
void display();
void frontsreen();
void drawstring();
void setFont();
void myMouse();
void update();
void control();
void helpscreen();
GLint a=300,b=-300,flag=0,traffic_regulator=1,control_keyl,control_keyr;
GLfloat red=0,blue=1,green=.3;

GLfloat p=0,q=0,r=0;
//-----------
void *currentfont;

void setFont(void *font)
{
```

```
	currentfont=font;
}

void drawstring(float x,float y,float z,char *string)
{
	char *c;
	glRasterPos3f(x,y,z);

	for(c=string;*c!='\0';c++)
	{	glColor3f(0.0,0.0,0.0);
		glutBitmapCharacter(currentfont,*c);
	}
}

void frontscreen(void)
{
setFont(GLUT_BITMAP_TIMES_ROMAN_24);
glClearColor(0.15,0.1,0.01,0);/*background for cover page*/
glClear(GL_COLOR_BUFFER_BIT);
glColor3f(1,0,0);
drawstring(450.0,700.0,0.0,"VELLORE INSTITUTE OF TECHNOLOGY ");
glColor3f(0.7,0,1);
drawstring(330,650,0.0,"MTECH INTEGRATED SOFTWARE ENGINEERING");
glColor3f(1,0.5,0);
drawstring(530,600,0.0,"A PROJECT ON");
glColor3f(1,0,0);
drawstring(360,500,0.0,"PARKING SYSTEM USING OPENGL");
glColor3f(1,0.5,0);
drawstring(200,400,0.0,"BY:");
glColor3f(1,1,1);
```

```
drawstring(100,300,0.0,"M SAI DEEPU -  16MIS0133");
glColor3f(1,1,1);
drawstring(100,240,0.0,"M NIKHIL - 16MIS0256");
glColor3f(1,0.5,0);
drawstring(980,400,0.0,"GUIDE:");
glColor3f(1,1,1);
drawstring(930,300,0.0,"GUNASEKARAN G");
glColor3f(1,1,1);
drawstring(543,100,0.0,"PRESS ENTER TO START");
glFlush();
}

void helpscreen()
{
setFont(GLUT_BITMAP_TIMES_ROMAN_24);
glClearColor(1.0,1.0,0,0);/*background for cover page*/
glClear(GL_COLOR_BUFFER_BIT);
glColor3f(0,1,0);
drawstring(550.0,700.0,0.0,"TIPS");
glColor3f(1,0,0);
drawstring(650.0,700.0,0.0,"AND");
glColor3f(0,0,1);
drawstring(750.0,700.0,0.0,"TRICKS");
glColor3f(0.5,0.1,0.2);
drawstring(350.0,640.0,0.0,"Stop the traffic (Red Light)                    MOUSE LEFT
CLICK");
glColor3f(0.5,0.1,0.3);
drawstring(350.0,540.0,0.0,"Yellow Signal                                    MOUSE
RIGHT BUTTON (HOLD ON)");
glColor3f(0.5,0.1,0.4);
```

```
drawstring(350.0,440.0,0.0,"Green Signal                                    MOUSE
RIGHT BUTTON (RELEASE)");
glColor3f(0.4,0.1,0.5);
drawstring(350.0,340.0,0.0,"Allow vehicles to MOVE left to right            PRESS 'L'");
glColor3f(0.5,0.1,0.6);
drawstring(350.0,240.0,0.0,"Allow vehicles to MOVE right to left            PRESS 'R'");
glColor3f(0.5,0.1,0.7);
drawstring(350.0,140.0,0.0,"Speed up the vehicles                           PRESS 'S'");
glColor3f(0.5,0.1,0.8);
drawstring(350.0,90.0,0.0,"Help                                            PRESS 'H'");
glColor3f(0.5,0.1,0.9);
drawstring(350.0,40.0,0.0,"Escape                                          PRESS
'ENTER'");
glFlush();
}

void control()
{
        if(control_keyl!='l'||control_keyr!='r')
        {
        if(control_keyl=='l')
    a=a+6;
        if(control_keyr=='r')
         b=b-6;
        }

}

void myKeyboard( unsigned char key, int x, int y )

{
```

```
switch(key)
{
	case 13:
		if(flag==1)
		{
			flag=2;
			mydisplay();
		}
		if(flag==0) //Ascii of 'enter' key is 13
		{
	flag=1;mydisplay();
		}

	break;
	case 'l':control_keyl=key;
		p=0;q=0;r=1;
	break;
	case 'r':control_keyr=key;
		p=0;q=0;r=1;
	break;
	case 's':mydisplay();
	break;
	case 'h':flag=1;mydisplay();
		break;
	default:break;
}

}

void myMouse(int button,int state,int x,int y)
```

```
{
        if(button==GLUT_LEFT_BUTTON && state==GLUT_DOWN)
        {
        traffic_regulator=0;
        p=1;
        q=0;
        r=0;
        }

        if(button==GLUT_RIGHT_BUTTON && state==GLUT_DOWN)
        {
        traffic_regulator=0;
        p=0;
        q=1;
        r=0;
        }
        if(button==GLUT_RIGHT_BUTTON && state==GLUT_UP)
        {
        traffic_regulator=1;
        p=0;
        q=0;
        r=1;
        }
        glutPostRedisplay();




}
```

```
void mydisplay(void)
{
glClear(GL_COLOR_BUFFER_BIT);
if(flag==0)
frontscreen ();
if(flag==1)
        helpscreen();
if(flag==2)
display();
glutSwapBuffers();
}

void update(int value)
{
a=a-6;
b=b+6;
control();
/*making day to night*/
//if(blue!=0&&green!=0)
//{blue-=.004;green-=.004;
//}

glutPostRedisplay();
}

void display(void)
{
        if(traffic_regulator)
glutTimerFunc(50,update,0);
glClear(GL_COLOR_BUFFER_BIT);
```

```
glClearColor(red,green,blue,0);/*back ground for sky*/
road();
bus();
signal();
car();
car2();

glFlush();
}

void road()
{
glPushMatrix();
glScaled(40.0,40.0,0.0);
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
//straight road
glVertex2f(0,5);
glVertex2f(40,5);
glVertex2f(40,10);
glVertex2f(0,10);
glEnd();
//green edge
glBegin(GL_POLYGON);
glColor3f(0.1,0.2,0.1);
glVertex2f(0,5);
glVertex2f(40,5);
glVertex2f(40,4);
glVertex2f(0,4);
glEnd();
```

```
//cross road
glColor3f(0.1,0.1,0.1);
glBegin(GL_POLYGON);
glVertex2f(10,10);
glVertex2f(15,10);
glVertex2f(0,40);
glVertex2f(4,40);
glEnd();
glPopMatrix();
}

void signal()
{
glPushMatrix();
glScaled(40.0,40.0,0.0);
//stand
glColor3f(0.1,0.2,0.1);
glBegin(GL_POLYGON);
glVertex2f(15,7);
glVertex2f(15,8);
glVertex2f(18,8);
glVertex2f(18,7);
glEnd();
//pole
glBegin(GL_POLYGON);
glVertex2f(16,7);
glVertex2f(17,8);
glVertex2f(17,15);
glVertex2f(16,15);
glEnd();
```

```
//board
glBegin(GL_POLYGON);
glVertex2f(15.5,15);
glVertex2f(17.5,15);
glVertex2f(17.5,10);
glVertex2f(15.5,10);
glEnd();
//red
glColor3f(p,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(16,14.5);
glVertex2f(17,14.5);
glVertex2f(17,14);
glVertex2f(16,14);
glEnd();
//yellow
glColor3f(q,q,0.0);
glBegin(GL_POLYGON);
glVertex2f(16,13.5);
glVertex2f(17,13.5);
glVertex2f(17,13);
glVertex2f(16,13);
glEnd();
//green
glColor3f(0.0,r,0.0);
glBegin(GL_POLYGON);
glVertex2f(16,12.5);
glVertex2f(17,12.5);
glVertex2f(17,12);
glVertex2f(16,12);
```

```
glEnd();
glPopMatrix();
}

void bus()
{
glPushMatrix();
glTranslated(a,50.0,0.0);
glScaled(40.0,40.0,0.0);
glColor3f(0.5,0.0,0.0);
//bus out line
glBegin(GL_POLYGON);
glVertex2f(25,8);
glVertex2f(25,9.5);
glVertex2f(26,11);
glVertex2f(32,11);
glVertex2f(32,8);
glEnd();
//window frame
glColor3f(0,0.1,1);
glBegin(GL_POLYGON);
glVertex2f(26.1,9.5);
glVertex2f(26.1,10.5);
glVertex2f(31.8,10.5);
glVertex2f(31.8,9.5);
glEnd();
//Doors
glColor3f(0,0.8,1);
glBegin(GL_POLYGON);
glVertex2f(26.2,9);
```

```
glVertex2f(26.2,10.4);
glVertex2f(27.7,10.4);
glVertex2f(27.7,9);
glEnd();

//entry door
glColor3f(1,1,1);
glBegin(GL_POLYGON);
glVertex2f(27,8.4);
glVertex2f(27,10.4);
glVertex2f(27.7,10.4);
glVertex2f(27.7,8.4);
glEnd();

//small windows
glColor3f(0,1,1);
glBegin(GL_POLYGON);
glVertex2f(27.8,9.6);
glVertex2f(27.8,10.4);
glVertex2f(29,10.4);
glVertex2f(29,9.6);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(29.1,9.6);
glVertex2f(29.1,10.4);
glVertex2f(30.2,10.4);
glVertex2f(30.2,9.6);
glEnd();
glBegin(GL_POLYGON);
glVertex2f(30.3,9.6);
```

```
glVertex2f(30.3,10.4);
glVertex2f(31.7,10.4);
glVertex2f(31.7,9.6);
glEnd();


//driver window
glColor3f(0,0.8,1);
glBegin(GL_POLYGON);
glPushMatrix();//front person
//glScaled(10.0,10.0,10.0);
glColor3f(0.0,0.0,0.0);
glVertex2f(25,9.5);
glVertex2f(26,11);
glVertex2f(26,9.5);
glEnd();
glPopMatrix();
//tyre
glPushMatrix();//front tyre
glTranslated(a+970,320,0.0);
glScaled(20.0,20.0,0.0);
glColor3f(0.0,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(3.0,2.5);
glVertex2f(3.0,2.6);
glVertex2f(3.15,3.1);
glVertex2f(3.2,3.2);
glVertex2f(3.3,3.35);
glVertex2f(3.4,3.4);
glVertex2f(3.5,3.45);
```

```
glVertex2f(3.6,3.55);
glVertex2f(3.7,3.6);
glVertex2f(3.8,3.63);
glVertex2f(4.0,3.65);
glVertex2f(4.2,3.7);
glVertex2f(4.4,3.7);
glVertex2f(4.6,3.65);
glVertex2f(4.8,3.55);
glVertex2f(5.0,3.45);
glVertex2f(5.1,3.4);
glVertex2f(5.2,3.25);
glVertex2f(5.3,3.2);
glVertex2f(5.4,3.0);
glVertex2f(5.5,2.5);

glVertex2f(5.45,2.15);
glVertex2f(5.4,1.9);
glVertex2f(5.35,1.8);
glVertex2f(5.2,1.6);
glVertex2f(5.0,1.5);
glVertex2f(4.9,1.4);
glVertex2f(4.7,1.3);
glVertex2f(4.6,1.27);
glVertex2f(4.4,1.25);
glVertex2f(4.0,1.25);
glVertex2f(3.9,1.3);
glVertex2f(3.75,1.35);
glVertex2f(3.6,1.4);
glVertex2f(3.45,1.55);
glVertex2f(3.3,1.7);
```

```
glVertex2f(3.2,1.8);
glVertex2f(3.1,2.2);
glEnd();
glPopMatrix();

glPushMatrix();//back tyre
glTranslated(a+1140,320,0.0);
glScaled(20.0,20.0,0.0);
glColor3f(0.0,0.0,0.0);
glBegin(GL_POLYGON);
glVertex2f(3.0,2.5);
glVertex2f(3.0,2.6);
glVertex2f(3.15,3.1);
glVertex2f(3.2,3.2);
glVertex2f(3.3,3.35);
glVertex2f(3.4,3.4);
glVertex2f(3.5,3.45);
glVertex2f(3.6,3.55);
glVertex2f(3.7,3.6);
glVertex2f(3.8,3.63);
glVertex2f(4.0,3.65);
glVertex2f(4.2,3.7);
glVertex2f(4.4,3.7);
glVertex2f(4.6,3.65);
glVertex2f(4.8,3.55);
glVertex2f(5.0,3.45);
glVertex2f(5.1,3.4);
glVertex2f(5.2,3.25);
glVertex2f(5.3,3.2);
glVertex2f(5.4,3.0);
```

```
glVertex2f(5.5,2.5);

glVertex2f(5.45,2.15);
glVertex2f(5.4,1.9);
glVertex2f(5.35,1.8);
glVertex2f(5.2,1.6);
glVertex2f(5.0,1.5);
glVertex2f(4.9,1.4);
glVertex2f(4.7,1.3);
glVertex2f(4.6,1.27);
glVertex2f(4.4,1.25);
glVertex2f(4.0,1.25);
glVertex2f(3.9,1.3);
glVertex2f(3.75,1.35);
glVertex2f(3.6,1.4);
glVertex2f(3.45,1.55);
glVertex2f(3.3,1.7);
glVertex2f(3.2,1.8);
glVertex2f(3.1,2.2);
glEnd();
glPopMatrix();
}

void car()
{
glPushMatrix(); //making color for outer line
glTranslated(b,190.0,0.0);
glScaled(20.0,20.0,0.0);
glColor3f(1.0,0.0,0.0);
glBegin(GL_POLYGON);
```

```
glVertex2f(2.5,2.5);
glVertex2f(3.0,3.5);
glVertex2f(3.5,3.75);
glVertex2f(4.0,4.0);
glVertex2f(4.5,4.0);
glVertex2f(5.0,3.75);
glVertex2f(5.5,3.5);
glVertex2f(5.75,3.0);
glVertex2f(6.0,2.5);
glVertex2f(16.5,2.5);
glVertex2f(16.75,3.0);
glVertex2f(17.0,3.5);
glVertex2f(17.5,3.75);
glVertex2f(18.0,4.0);
glVertex2f(18.5,4.0);
glVertex2f(19.0,3.75);
glVertex2f(19.5,3.5);
glVertex2f(19.75,3.0);
glVertex2f(20.0,2.5);
glVertex2f(21.0,2.5);
glVertex2f(21.0,4.0);
glVertex2f(21.5,4.0);
glVertex2f(21.0,4.5);
glVertex2f(20.0,5.0);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.5);
glVertex2f(13.0,6.0);
glVertex2f(12.0,6.5);
glVertex2f(11.0,7.0);
glVertex2f(6.0,7.0);
```

```
glVertex2f(5.0,6.5);
glVertex2f(4.5,6.25);
glVertex2f(4.25,6.0);
glVertex2f(4.0,5.75);
glVertex2f(3.5,5.5);
glVertex2f(3.0,5.5);
glVertex2f(1.9,5.45);
glVertex2f(1.8,5.4);
glVertex2f(1.7,5.35);
glVertex2f(1.6,5.3);
glVertex2f(1.5,5.25);
glVertex2f(1.4,5.15);
glVertex2f(1.3,5.0);
glVertex2f(1.2,4.85);
glVertex2f(1.1,4.7);
glVertex2f(1.0,4.3);
glVertex2f(1.0,3.2);
glVertex2f(1.1,3.05);
glVertex2f(1.2,2.9);
glVertex2f(1.3,2.9);
glVertex2f(1.4,2.75);
glVertex2f(1.5,2.65);
glVertex2f(1.6,2.6);
glVertex2f(1.7,2.55);
glVertex2f(1.8,2.5);
glVertex2f(1.9,2.45);
glVertex2f(2.0,2.5);
glEnd();

glColor3f(1.0,1.0,1.0); //color for outer window
```

```
glBegin(GL_POLYGON);
glVertex2f(5.0,5.0);
glVertex2f(14.0,5.0);
glVertex2f(11.5,6.5);
glVertex2f(10.5,6.75);
glVertex2f(7.0,6.75);
glEnd();

glColor3f(0.0,0.0,0.0); //making outer line for car
glBegin(GL_LINE_LOOP);
glVertex2f(2.5,2.5);
glVertex2f(3.0,3.5);
glVertex2f(3.5,3.75);
glVertex2f(4.0,4.0);
glVertex2f(4.5,4.0);
glVertex2f(5.0,3.75);
glVertex2f(5.5,3.5);
glVertex2f(5.75,3.0);
glVertex2f(6.0,2.5);
glVertex2f(16.5,2.5);
glVertex2f(16.75,3.0);
glVertex2f(17.0,3.5);
glVertex2f(17.5,3.75);
glVertex2f(18.0,4.0);
glVertex2f(18.5,4.0);
glVertex2f(19.0,3.75);
glVertex2f(19.5,3.5);
glVertex2f(19.75,3.0);
glVertex2f(20.0,2.5);
glVertex2f(21.0,2.5);
```

```
glVertex2f(21.0,4.0);
glVertex2f(21.5,4.0);
glVertex2f(21.0,4.5);
glVertex2f(20.0,5.0);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.5);
glVertex2f(13.0,6.0);
glVertex2f(12.0,6.5);
glVertex2f(11.0,7.0);
glVertex2f(6.0,7.0);
glVertex2f(5.0,6.5);
glVertex2f(4.5,6.25);
glVertex2f(4.25,6.0);
glVertex2f(4.0,5.75);
glVertex2f(3.5,5.5);
glVertex2f(3.0,5.5);
glVertex2f(1.9,5.45);
glVertex2f(1.8,5.4);
glVertex2f(1.7,5.35);
glVertex2f(1.6,5.3);
glVertex2f(1.5,5.25);
glVertex2f(1.4,5.15);
glVertex2f(1.3,5.0);
glVertex2f(1.2,4.85);
glVertex2f(1.1,4.7);
glVertex2f(1.0,4.3);
glVertex2f(1.0,3.2);
glVertex2f(1.1,3.05);
glVertex2f(1.2,2.9);
glVertex2f(1.3,2.9);
```

```
glVertex2f(1.4,2.75);
glVertex2f(1.5,2.65);
glVertex2f(1.6,2.6);
glVertex2f(1.7,2.55);
glVertex2f(1.8,2.5);
glVertex2f(1.9,2.45);
glVertex2f(2.0,2.5);
glEnd();

glColor3f(0.0,0.0,0.0);
glBegin(GL_LINE_LOOP); //outer line for design a car
glVertex2f(8.0,3.0);
glVertex2f(16.0,3.0);
glVertex2f(16.5,3.5);
glVertex2f(17.0,4.0);
glVertex2f(16.5,4.25);
glVertex2f(16.0,4.5);
glVertex2f(15.0,4.5);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.0);
glVertex2f(11.5,6.5);
glVertex2f(10.5,6.75);
glVertex2f(7.0,6.75);
glVertex2f(5.0,5.0);
glVertex2f(7.0,5.0);
glVertex2f(6.5,4.5);
glEnd();


glBegin(GL_LINES); //connecting outer line
```

```
glVertex2d(7.0,5.0);
glVertex2d(15.0,5.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,4.0);
glVertex2d(17.0,4.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,3.5);
glVertex2d(16.5,3.5);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,5.0);
glVertex2d(14.0,3.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(12.0,5.0);
glVertex2d(12.0,6.2);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
```

```
glVertex2d(7.0,5.0);
glVertex2d(7.0,6.7);
glEnd();

glBegin(GL_POLYGON); //drawing a back tyre
glVertex2f(3.0,2.5);
glVertex2f(3.0,2.6);
glVertex2f(3.15,3.1);
glVertex2f(3.2,3.2);
glVertex2f(3.3,3.35);
glVertex2f(3.4,3.4);
glVertex2f(3.5,3.45);
glVertex2f(3.6,3.55);
glVertex2f(3.7,3.6);
glVertex2f(3.8,3.63);
glVertex2f(4.0,3.65);
glVertex2f(4.2,3.7);
glVertex2f(4.4,3.7);
glVertex2f(4.6,3.65);
glVertex2f(4.8,3.55);
glVertex2f(5.0,3.45);
glVertex2f(5.1,3.4);
glVertex2f(5.2,3.25);
glVertex2f(5.3,3.2);
glVertex2f(5.4,3.0);
glVertex2f(5.5,2.5);

glVertex2f(5.45,2.15);
glVertex2f(5.4,1.9);
glVertex2f(5.35,1.8);
```

```
glVertex2f(5.2,1.6);
glVertex2f(5.0,1.5);
glVertex2f(4.9,1.4);
glVertex2f(4.7,1.3);
glVertex2f(4.6,1.27);
glVertex2f(4.4,1.25);
glVertex2f(4.0,1.25);
glVertex2f(3.9,1.3);
glVertex2f(3.75,1.35);
glVertex2f(3.6,1.4);
glVertex2f(3.45,1.55);
glVertex2f(3.3,1.7);
glVertex2f(3.2,1.8);
glVertex2f(3.1,2.2);
glEnd();


glBegin(GL_POLYGON); //drawing front tyre
glVertex2f(17.0,2.5);
glVertex2f(17.0,2.6);
glVertex2f(17.15,3.1);
glVertex2f(17.2,3.2);
glVertex2f(17.3,3.35);
glVertex2f(17.4,3.4);
glVertex2f(17.5,3.45);
glVertex2f(17.6,3.55);
glVertex2f(17.7,3.6);
glVertex2f(17.8,3.63);
glVertex2f(18.0,3.65);
glVertex2f(18.2,3.7);
```

```
glVertex2f(18.4,3.7);
glVertex2f(18.6,3.65);
glVertex2f(18.8,3.55);
glVertex2f(19.0,3.45);
glVertex2f(19.1,3.4);
glVertex2f(19.2,3.25);
glVertex2f(19.3,3.2);
glVertex2f(19.4,3.0);

glVertex2f(19.5,2.5);
glVertex2f(19.45,2.15);
glVertex2f(19.4,1.9);
glVertex2f(19.35,1.8);
glVertex2f(19.2,1.6);
glVertex2f(19.0,1.5);
glVertex2f(18.9,1.4);
glVertex2f(18.7,1.3);
glVertex2f(18.6,1.27);
glVertex2f(18.4,1.25);
glVertex2f(18.0,1.25);
glVertex2f(17.9,1.3);
glVertex2f(17.75,1.35);
glVertex2f(17.6,1.4);
glVertex2f(17.45,1.55);
glVertex2f(17.3,1.7);
glVertex2f(17.2,1.8);
glVertex2f(17.1,2.2);
glEnd();
glPopMatrix();
}
```

```
void car2()
{
glPushMatrix(); //making color for outer line
glTranslated(b-2000,190.0,0.0);
glScaled(20.0,20.0,0.0);
glColor3f(1.0,1.0,0.4);
glBegin(GL_POLYGON);
glVertex2f(2.5,2.5);
glVertex2f(3.0,3.5);
glVertex2f(3.5,3.75);
glVertex2f(4.0,4.0);
glVertex2f(4.5,4.0);
glVertex2f(5.0,3.75);
glVertex2f(5.5,3.5);
glVertex2f(5.75,3.0);
glVertex2f(6.0,2.5);
glVertex2f(16.5,2.5);
glVertex2f(16.75,3.0);
glVertex2f(17.0,3.5);
glVertex2f(17.5,3.75);
glVertex2f(18.0,4.0);
glVertex2f(18.5,4.0);
glVertex2f(19.0,3.75);
glVertex2f(19.5,3.5);
glVertex2f(19.75,3.0);
glVertex2f(20.0,2.5);
glVertex2f(21.0,2.5);
glVertex2f(21.0,4.0);
glVertex2f(21.5,4.0);
glVertex2f(21.0,4.5);
```

```
glVertex2f(20.0,5.0);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.5);
glVertex2f(13.0,6.0);
glVertex2f(12.0,6.5);
glVertex2f(11.0,7.0);
glVertex2f(6.0,7.0);
glVertex2f(5.0,6.5);
glVertex2f(4.5,6.25);
glVertex2f(4.25,6.0);
glVertex2f(4.0,5.75);
glVertex2f(3.5,5.5);
glVertex2f(3.0,5.5);
glVertex2f(1.9,5.45);
glVertex2f(1.8,5.4);
glVertex2f(1.7,5.35);
glVertex2f(1.6,5.3);
glVertex2f(1.5,5.25);
glVertex2f(1.4,5.15);
glVertex2f(1.3,5.0);
glVertex2f(1.2,4.85);
glVertex2f(1.1,4.7);
glVertex2f(1.0,4.3);
glVertex2f(1.0,3.2);
glVertex2f(1.1,3.05);
glVertex2f(1.2,2.9);
glVertex2f(1.3,2.9);
glVertex2f(1.4,2.75);
glVertex2f(1.5,2.65);
glVertex2f(1.6,2.6);
```

```
glVertex2f(1.7,2.55);
glVertex2f(1.8,2.5);
glVertex2f(1.9,2.45);
glVertex2f(2.0,2.5);
glEnd();

glColor3f(1.0,1.0,1.0); //color for outer window
glBegin(GL_POLYGON);
glVertex2f(5.0,5.0);
glVertex2f(14.0,5.0);
glVertex2f(11.5,6.5);
glVertex2f(10.5,6.75);
glVertex2f(7.0,6.75);
glEnd();

glColor3f(0.0,0.0,0.0); //making outer line for car
glBegin(GL_LINE_LOOP);
glVertex2f(2.5,2.5);
glVertex2f(3.0,3.5);
glVertex2f(3.5,3.75);
glVertex2f(4.0,4.0);
glVertex2f(4.5,4.0);
glVertex2f(5.0,3.75);
glVertex2f(5.5,3.5);
glVertex2f(5.75,3.0);
glVertex2f(6.0,2.5);
glVertex2f(16.5,2.5);
glVertex2f(16.75,3.0);
glVertex2f(17.0,3.5);
glVertex2f(17.5,3.75);
```

```
glVertex2f(18.0,4.0);
glVertex2f(18.5,4.0);
glVertex2f(19.0,3.75);
glVertex2f(19.5,3.5);
glVertex2f(19.75,3.0);
glVertex2f(20.0,2.5);
glVertex2f(21.0,2.5);
glVertex2f(21.0,4.0);
glVertex2f(21.5,4.0);
glVertex2f(21.0,4.5);
glVertex2f(20.0,5.0);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.5);
glVertex2f(13.0,6.0);
glVertex2f(12.0,6.5);
glVertex2f(11.0,7.0);
glVertex2f(6.0,7.0);
glVertex2f(5.0,6.5);
glVertex2f(4.5,6.25);
glVertex2f(4.25,6.0);
glVertex2f(4.0,5.75);
glVertex2f(3.5,5.5);
glVertex2f(3.0,5.5);
glVertex2f(1.9,5.45);
glVertex2f(1.8,5.4);
glVertex2f(1.7,5.35);
glVertex2f(1.6,5.3);
glVertex2f(1.5,5.25);
glVertex2f(1.4,5.15);
glVertex2f(1.3,5.0);
```

```
glVertex2f(1.2,4.85);
glVertex2f(1.1,4.7);
glVertex2f(1.0,4.3);
glVertex2f(1.0,3.2);
glVertex2f(1.1,3.05);
glVertex2f(1.2,2.9);
glVertex2f(1.3,2.9);
glVertex2f(1.4,2.75);
glVertex2f(1.5,2.65);
glVertex2f(1.6,2.6);
glVertex2f(1.7,2.55);
glVertex2f(1.8,2.5);
glVertex2f(1.9,2.45);
glVertex2f(2.0,2.5);
glEnd();

glColor3f(0.0,0.0,0.0);
glBegin(GL_LINE_LOOP); //outer line for design a car
glVertex2f(8.0,3.0);
glVertex2f(16.0,3.0);
glVertex2f(16.5,3.5);
glVertex2f(17.0,4.0);
glVertex2f(16.5,4.25);
glVertex2f(16.0,4.5);
glVertex2f(15.0,4.5);
glVertex2f(15.0,5.0);
glVertex2f(14.0,5.0);
glVertex2f(11.5,6.5);
glVertex2f(10.5,6.75);
glVertex2f(7.0,6.75);
```

```
glVertex2f(5.0,5.0);
glVertex2f(7.0,5.0);
glVertex2f(6.5,4.5);
glEnd();


glBegin(GL_LINES); //connecting outer line
glVertex2d(7.0,5.0);
glVertex2d(15.0,5.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,4.0);
glVertex2d(17.0,4.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,3.5);
glVertex2d(16.5,3.5);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(15.0,5.0);
glVertex2d(14.0,3.0);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
```

```
glBegin(GL_LINES);
glVertex2d(12.0,5.0);
glVertex2d(12.0,6.2);
glEnd();

glColor3f(0.0,0.0,0.0); //connecting outer line
glBegin(GL_LINES);
glVertex2d(7.0,5.0);
glVertex2d(7.0,6.7);
glEnd();

glBegin(GL_POLYGON); //drawing a back tyre
glVertex2f(3.0,2.5);
glVertex2f(3.0,2.6);
glVertex2f(3.15,3.1);
glVertex2f(3.2,3.2);
glVertex2f(3.3,3.35);
glVertex2f(3.4,3.4);
glVertex2f(3.5,3.45);
glVertex2f(3.6,3.55);
glVertex2f(3.7,3.6);
glVertex2f(3.8,3.63);
glVertex2f(4.0,3.65);
glVertex2f(4.2,3.7);
glVertex2f(4.4,3.7);
glVertex2f(4.6,3.65);
glVertex2f(4.8,3.55);
glVertex2f(5.0,3.45);
glVertex2f(5.1,3.4);
glVertex2f(5.2,3.25);
```

```
glVertex2f(5.3,3.2);
glVertex2f(5.4,3.0);
glVertex2f(5.5,2.5);

glVertex2f(5.45,2.15);
glVertex2f(5.4,1.9);
glVertex2f(5.35,1.8);
glVertex2f(5.2,1.6);
glVertex2f(5.0,1.5);
glVertex2f(4.9,1.4);
glVertex2f(4.7,1.3);
glVertex2f(4.6,1.27);
glVertex2f(4.4,1.25);
glVertex2f(4.0,1.25);
glVertex2f(3.9,1.3);
glVertex2f(3.75,1.35);
glVertex2f(3.6,1.4);
glVertex2f(3.45,1.55);
glVertex2f(3.3,1.7);
glVertex2f(3.2,1.8);
glVertex2f(3.1,2.2);
glEnd();


glBegin(GL_POLYGON); //drawing front tyre
glVertex2f(17.0,2.5);
glVertex2f(17.0,2.6);
glVertex2f(17.15,3.1);
glVertex2f(17.2,3.2);
glVertex2f(17.3,3.35);
```

```
glVertex2f(17.4,3.4);
glVertex2f(17.5,3.45);
glVertex2f(17.6,3.55);
glVertex2f(17.7,3.6);
glVertex2f(17.8,3.63);
glVertex2f(18.0,3.65);
glVertex2f(18.2,3.7);
glVertex2f(18.4,3.7);
glVertex2f(18.6,3.65);
glVertex2f(18.8,3.55);
glVertex2f(19.0,3.45);
glVertex2f(19.1,3.4);
glVertex2f(19.2,3.25);
glVertex2f(19.3,3.2);
glVertex2f(19.4,3.0);

glVertex2f(19.5,2.5);
glVertex2f(19.45,2.15);
glVertex2f(19.4,1.9);
glVertex2f(19.35,1.8);
glVertex2f(19.2,1.6);
glVertex2f(19.0,1.5);
glVertex2f(18.9,1.4);
glVertex2f(18.7,1.3);
glVertex2f(18.6,1.27);
glVertex2f(18.4,1.25);
glVertex2f(18.0,1.25);
glVertex2f(17.9,1.3);
glVertex2f(17.75,1.35);
glVertex2f(17.6,1.4);
```

```
glVertex2f(17.45,1.55);
glVertex2f(17.3,1.7);
glVertex2f(17.2,1.8);
glVertex2f(17.1,2.2);
glEnd();
glPopMatrix();
}

void myinit()
{

glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluOrtho2D(0.0,1346.0,0.0,728.0);
}

int main(int argc, char* argv[])
{
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
glutInitWindowSize(1346,728);
glutInitWindowPosition(0,0);
glutCreateWindow("Traffic signal");

/*call back functions*/
glutDisplayFunc(mydisplay);
glutKeyboardFunc(myKeyboard);
glutMouseFunc(myMouse);
```

```
myinit();
glutMainLoop();
}
```

Standards:

The source code used in the simulation of the traffic light system follows various standards of the programming. It contains the concepts of the

Polymorphism

Abstraction

Call by reference as well as

UI/UX standards

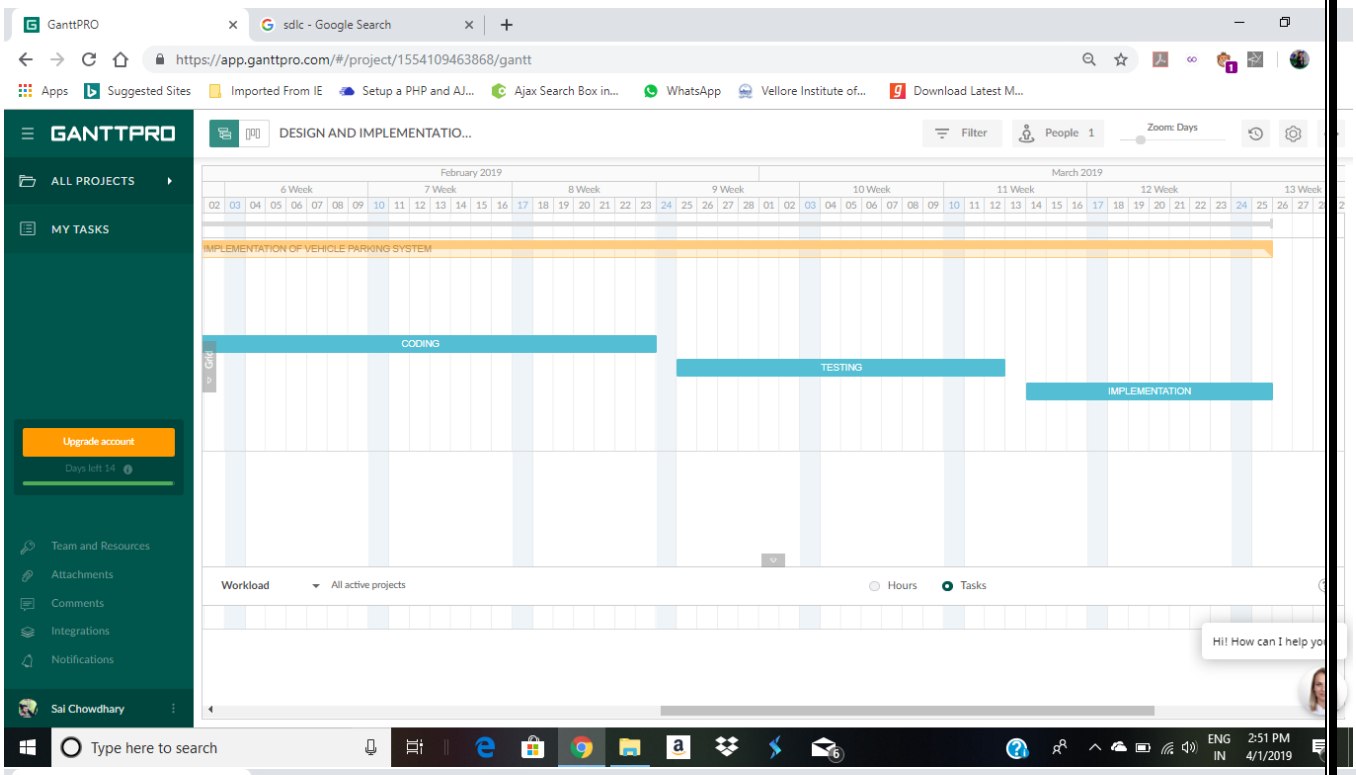
Constraints:

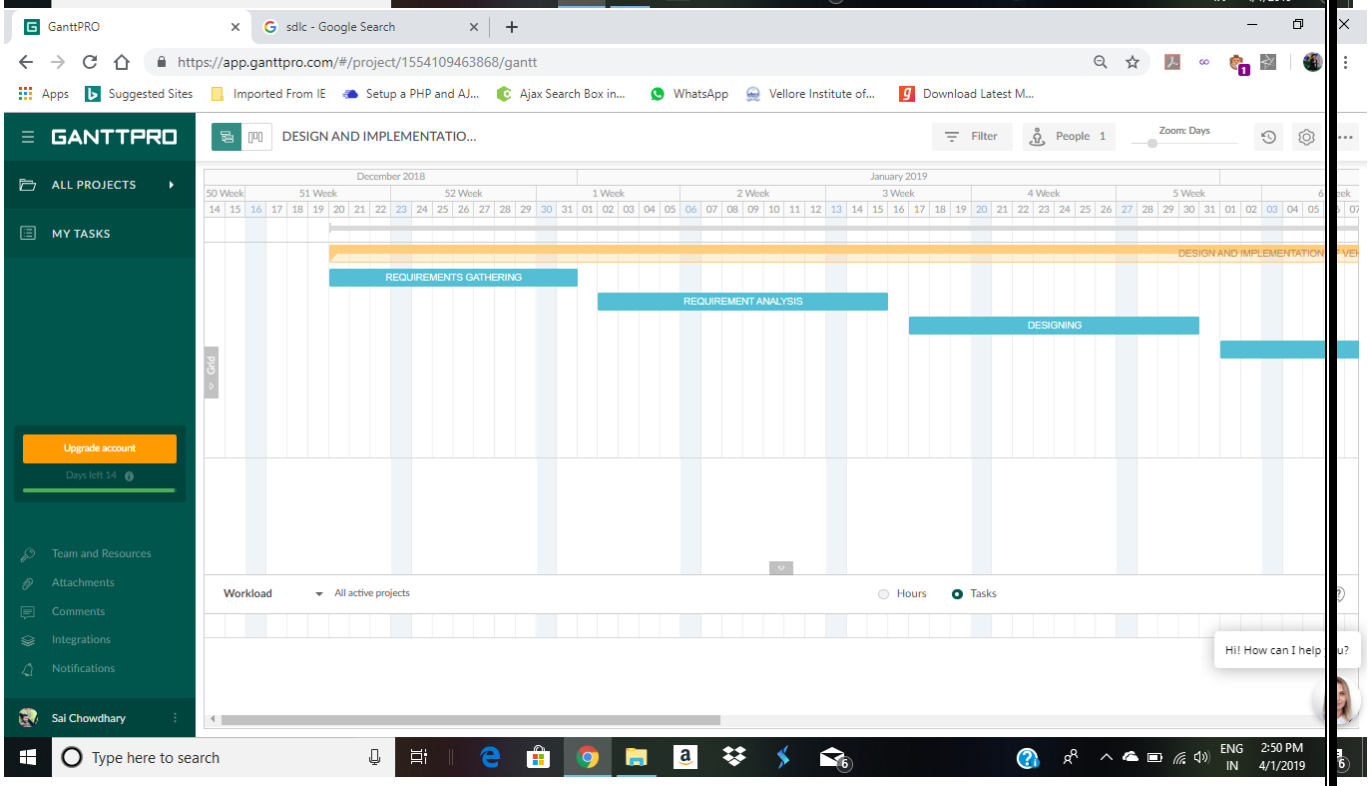
The major constraint in the system is that the system is not simulated under the concept of dynamic programming, in order to meet the condition of simulation of own choice.

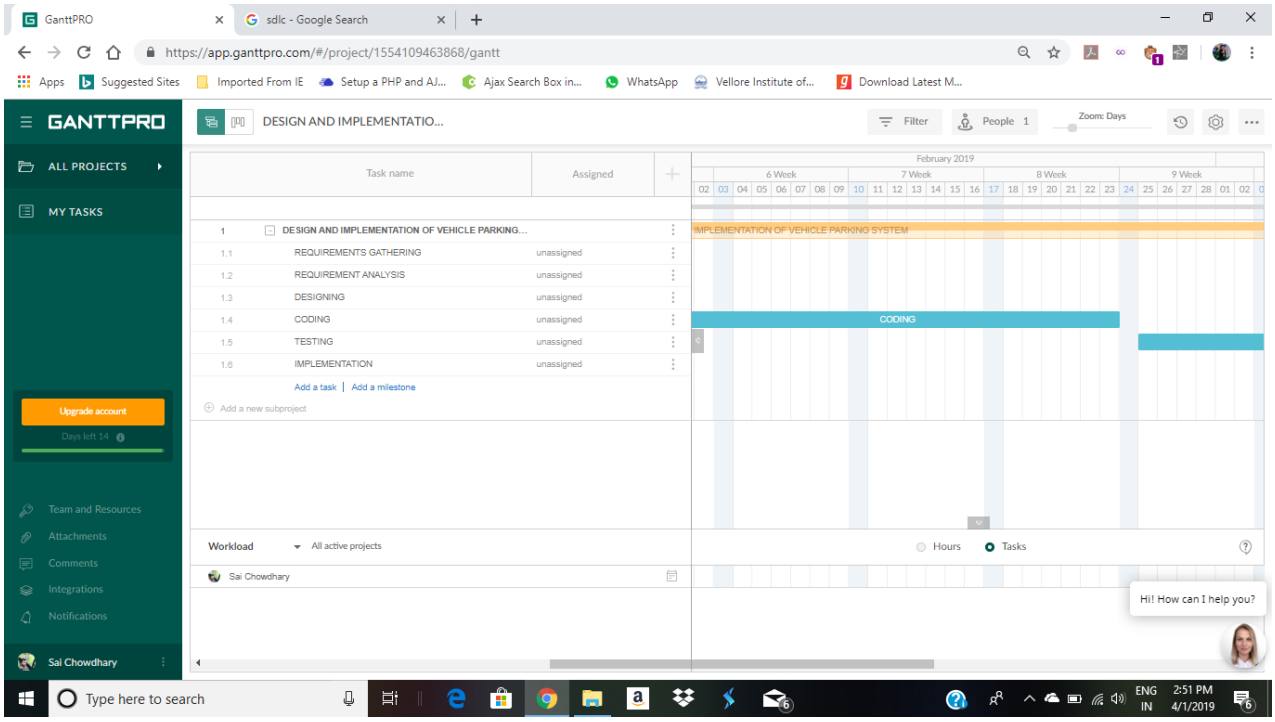
Higher the number of assets and objects in the frame, makes it hard and takes most of the efficiency of the processor effecting the other process to the maximum extent.

5. Project Schedule:

The project was scheduled in such a way that the work was distributed even throughout the schedule. Even though, the phase of system installation and setting up the environment took most of the time, the system finally got completed on time with a little deviation in the schedule.

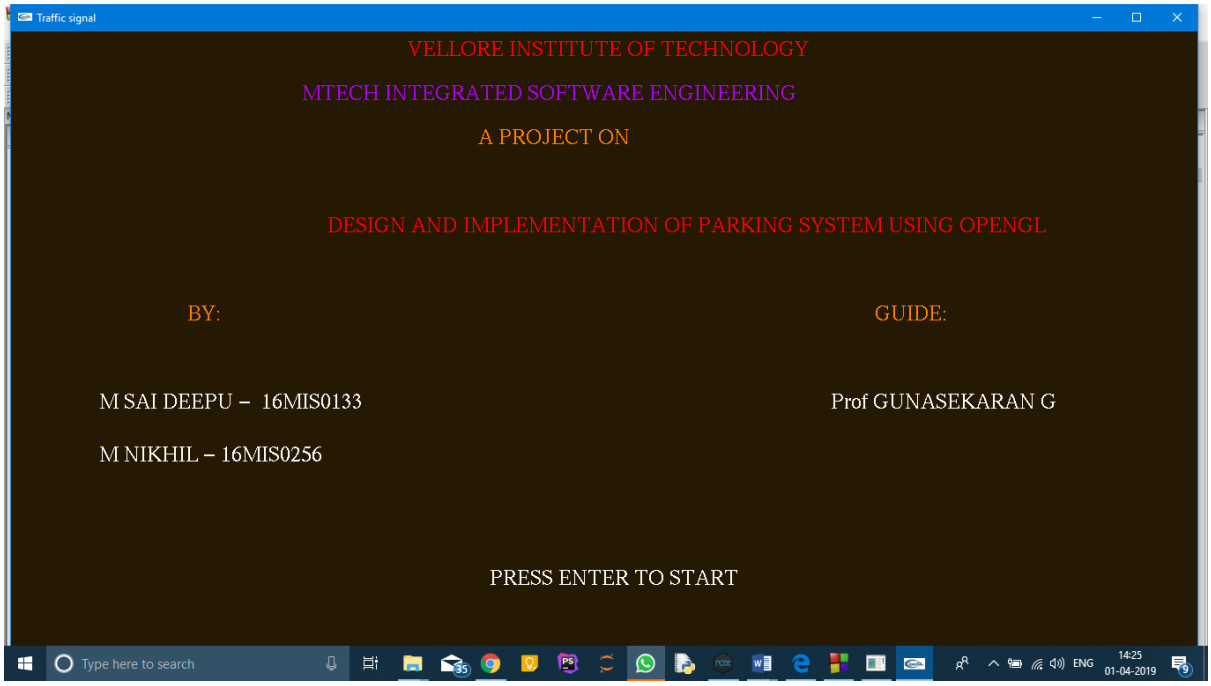

GanttPRO
sdlc - Google Search
https://app.ganttpro.com/#/project/1554109463868/gantt
GANTTPRO
DESIGN AND IMPLEMENTATIO...
Filter
People 1
Zoom: Days
ALL PROJECTS
MY TASKS
February 2019
March 2019
IMPLEMENTATION OF VEHICLE PARKING SYSTEM
CODING
TESTING
IMPLEMENTATION
Upgrade account
Days left 14
Team and Resources
Attachments
Comments
Integrations
Notifications
Sai Chowdhary
Workload
All active projects
Hours
Tasks
Hi! How can I help yo
Type here to search
ENG
IN
2:51 PM
4/1/2019


GanttPRO
sdlc - Google Search
https://app.ganttpro.com/#/project/1554109463868/gantt
GANTTPRO
DESIGN AND IMPLEMENTATIO...
Filter
People 1
Zoom: Days
ALL PROJECTS
MY TASKS
December 2018
January 2019
DESIGN AND IMPLEMENTATION
REQUIREMENTS GATHERING
REQUIREMENT ANALYSIS
DESIGNING
Upgrade account
Days left 14
Team and Resources
Attachments
Comments
Integrations
Notifications
Sai Chowdhary
Workload
All active projects
Hours
Tasks
Hi! How can I help
Type here to search
ENG
IN
2:50 PM
4/1/2019

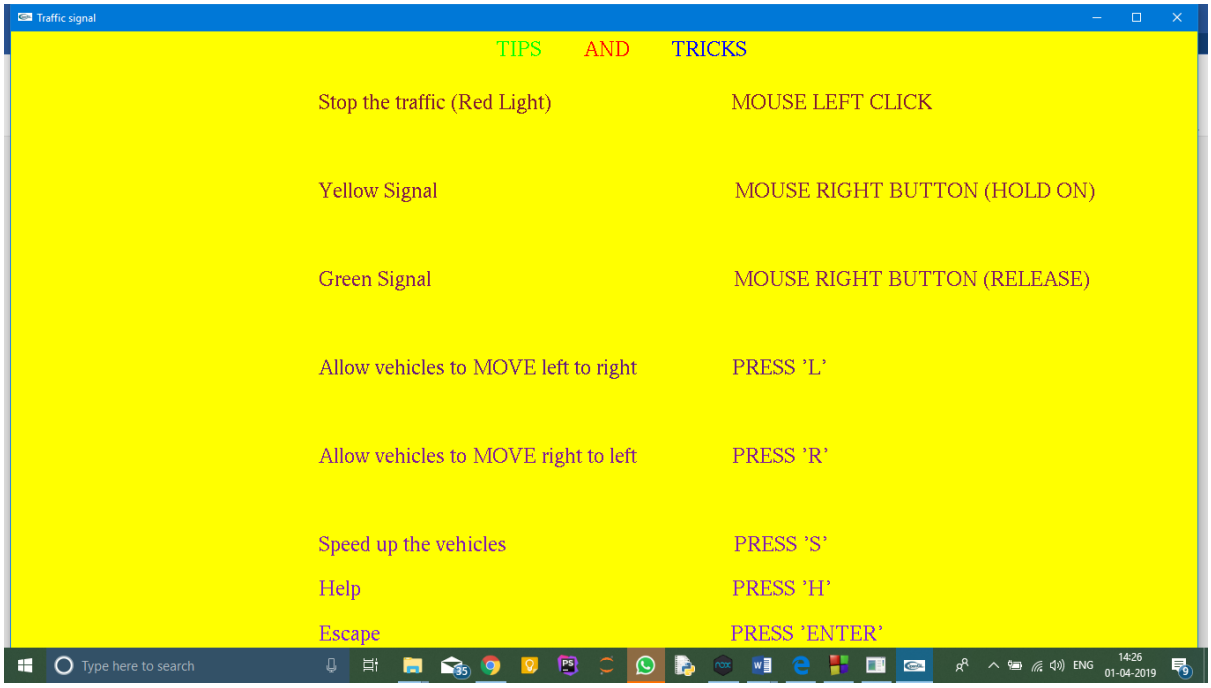


6. PROJECT DEMONSTRATION:

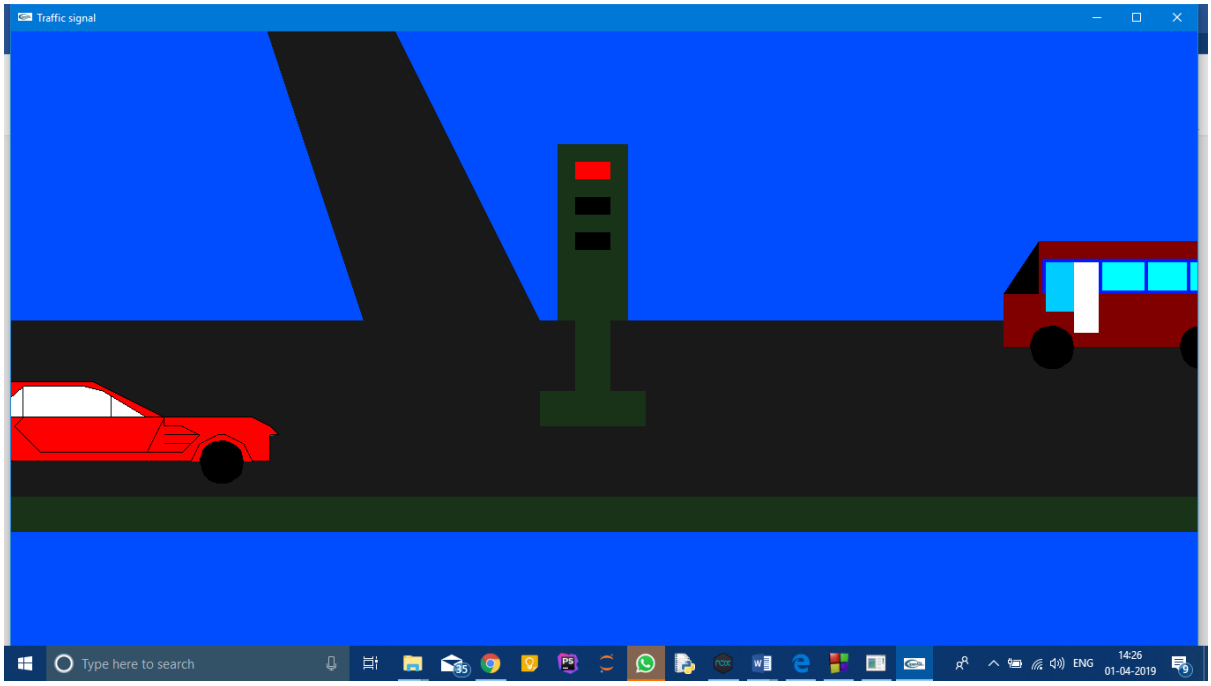
Step I: Project Initialization and Home Screen



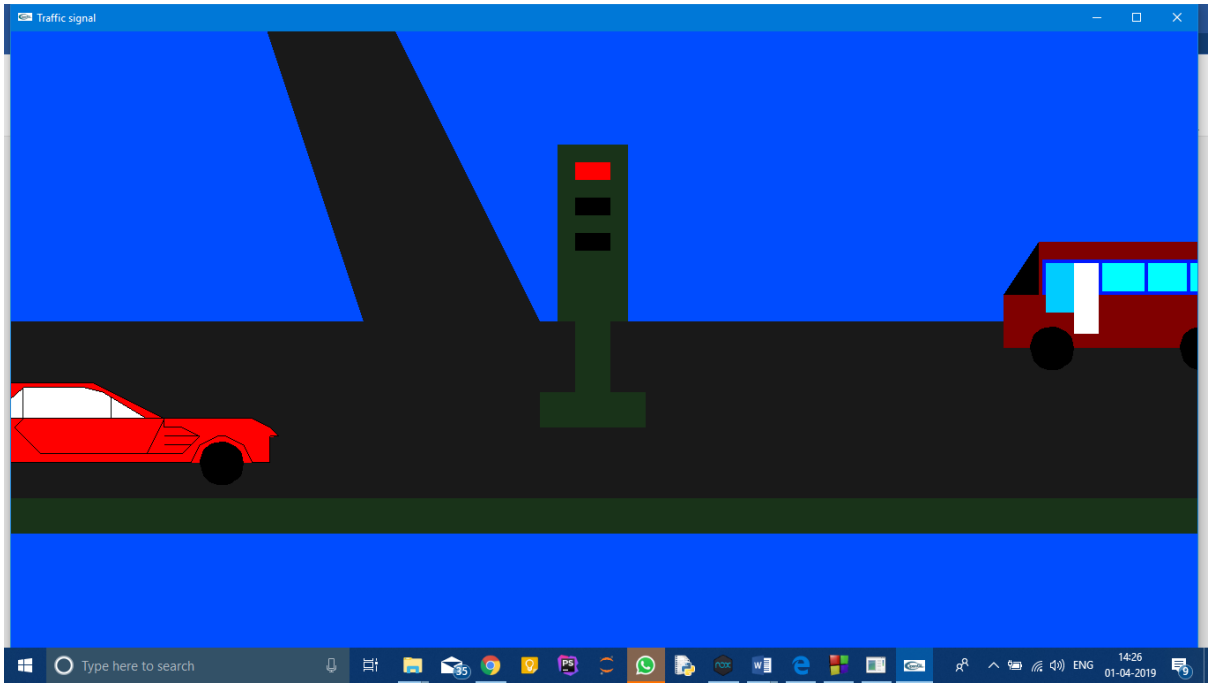
Step II: Help screen



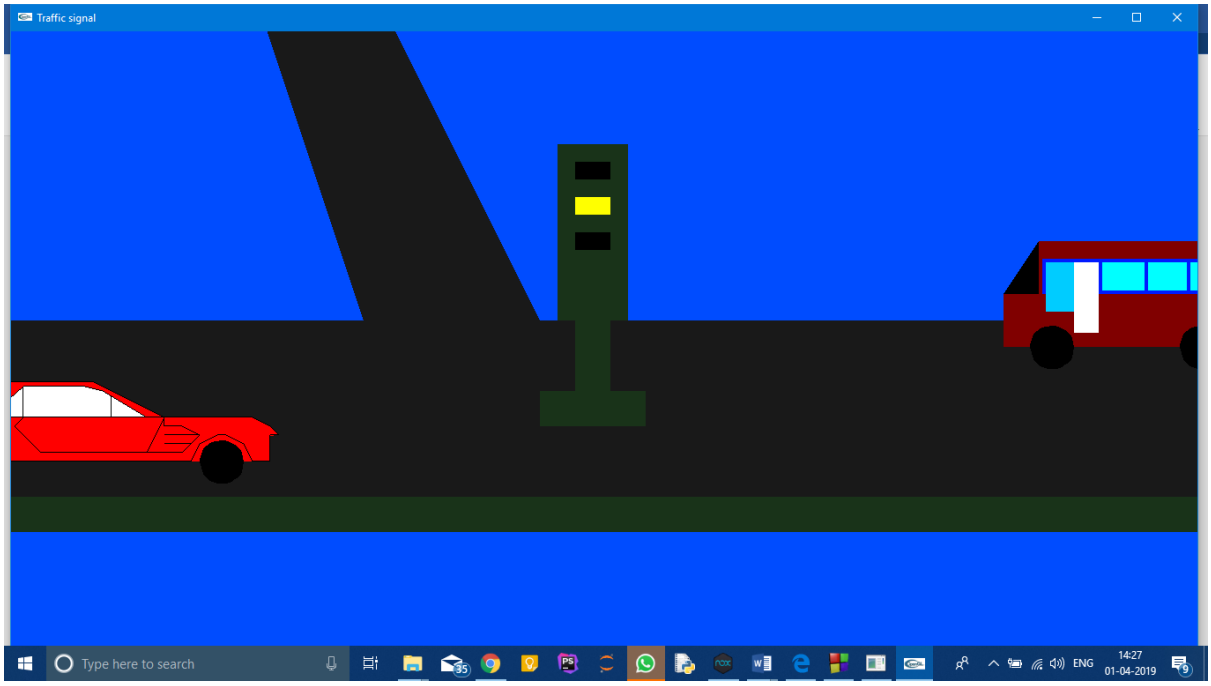
Step III: System Simulation



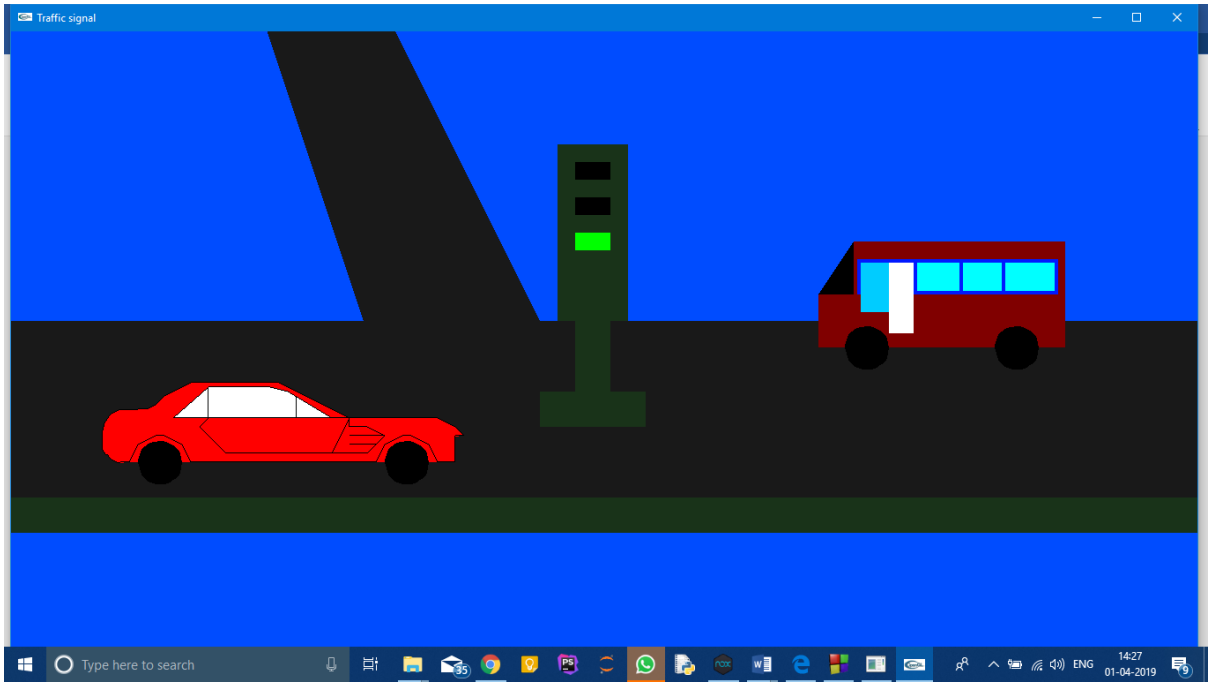
Step IV: System simulation (Red Light)



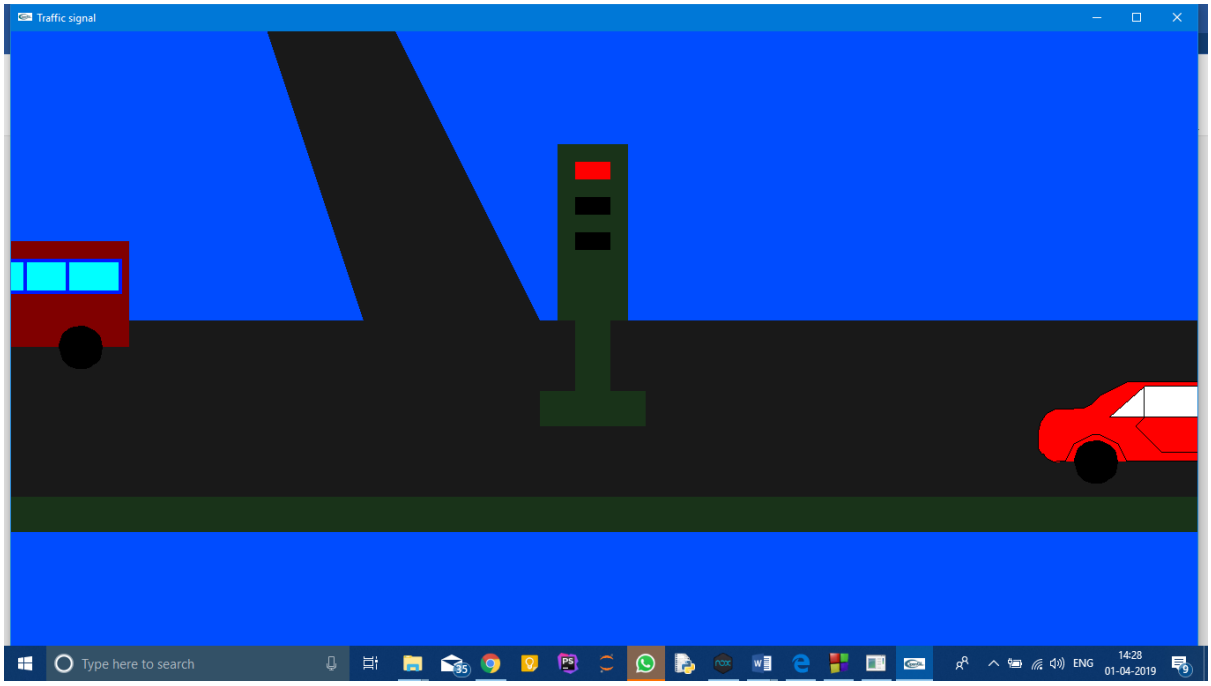
Step V: System simulation (Yellow Light)



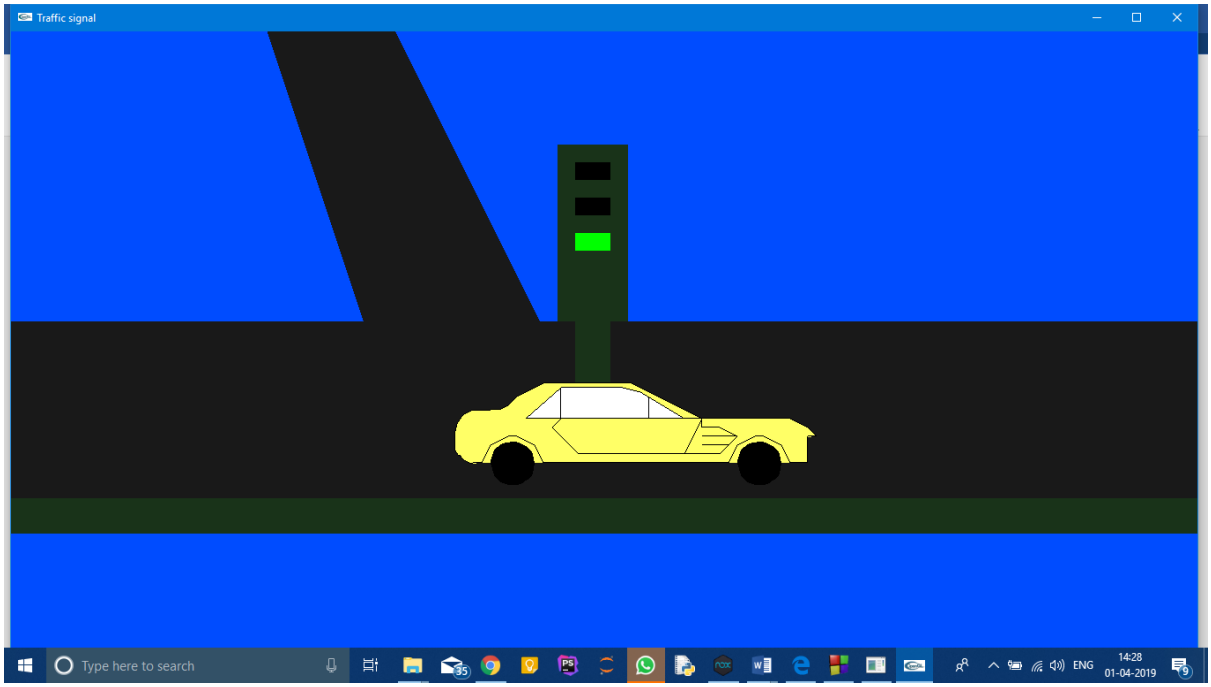
Step VI: Green Light



Step VII: Object Translation



Step VII: Object Speedup(Hit s)



7. RESULTS AND CONCLUSION

In this project the driver of the vehicle will be clearly understood that when to enter into the parking cellular. Traffic signal system is installed in the parking areas in any apartments, offices and other areas in metropolitan cities to reduce the traffic of vehicles entering into the

parking areas. This system will guide the drivers of the vehicles when to enter into the parking cellular. If suppose already a vehicle entered it instruct the other vehicle driver to wait for a while until the first entered vehicle is parked. This reduces the misunderstandings between the drivers in the parking cellular.

8. REFERENCES

[1] Ganesh Sharma, Nigidita Pradhan, Tanuj Karwa, Arun Kumar Singh.(2014) Design and Development of Automated Parking Slot. ISSN : 2230-7109 | ISSN : 2230-9543

[2] Ma, S., Jiang, H., Han, M., Xie, J., & Li, C. (2017). Research on automatic parking systems based on parking scene recognition. *IEEE Access*, *5*, 21901-21917.

[3] Mohmmed Ahmed Wang Guang Wei (2014) Study on Automated Car Parking System Based on Microcontroller (IJERT) Vol. 3 Issue 1, ISSN: 2278-0181

[4] Abdul Ahad, Zishan Raza Khan, Syed Aqeel Ahmad, (2016) Intelligent Parking System World Journal of Engineering and Technology, 2016, 4, 160-167

[5] Jayakshei Dadaji Bachhav1, Prof. Mechkul M. A.2 Smart Car Parking System International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056 Volume: 04 Issue: 06 | June -2017

[6] https://www.openglprojects.in