PUBLIC TRANSPORTATION OPTIMIZATION

PROJECT: SMART TRAFFIC CONTROL SYSTEM

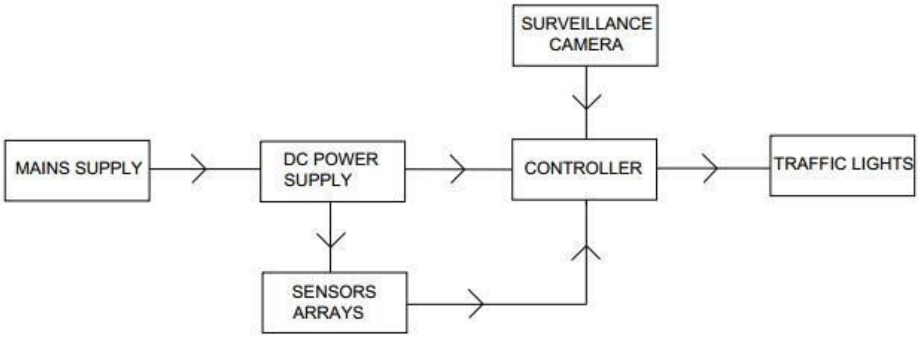
INTRODUCTION:

The signal is one of those elements that, for every computer, seem to be part of everyday life. Wherever the Traffic are more number of cars, there will probably be one of these systems are present. This project is an Arduino board Based Traffic Light system.

The traffic light system provides instructions to the users (drivers and pedestrians) by displaying lights of standard colour on four cross streets.

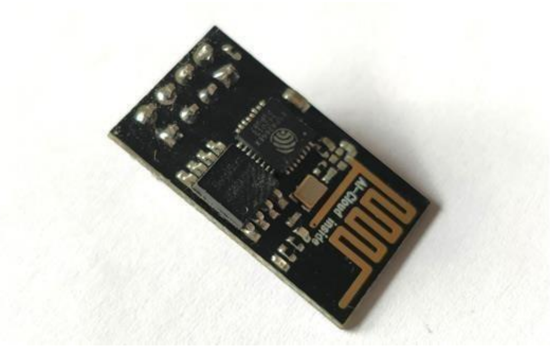
The colours used in traffic lights are Red, Yellow and Green for stop, slow and go respectively. The system is programmed to control the traffic lights for smooth and safe movement of vehicle traffic. The system consists of electro mechanical controllers with modern solid state computerized systems which has easy setup and maintenance.

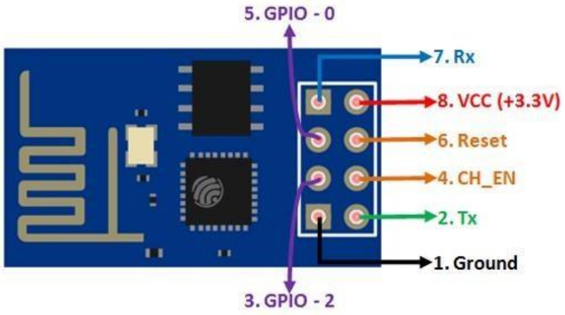
BLOCK DIAGRAM:



COMPONENTS EXPLANATION:

ESP8266 :





ESP8266 Pin Configuration :

Pin Number	**Pin Name**	**Alternate Name**	**Normally used for**	**Alternate purpose**
1	**Ground**	-	Connected to the ground of the circuit	-
2	**TX**	GPIO – 1	Connected to Rx pin of programmer/uC to upload program	Can act as a General purpose Input/output pin when not used as TX

3	**GPIO-2**	-	General purpose Input/output pin	-
4	**CH_EN**	-	Chip Enable – Active high	-
5	**GPIO - 0**	Flash	General purpose Input/output pin	Takes module into serial programming when held low during start up
6	**Reset**	-	Resets the module	-

7	**RX**	GPIO - 3	General purpose Input/output pin	Can act as a General purpose Input/output pin when not used as RX
8	**Vcc**	-	to Connect +3.3V only	

ESP8266-01 Features :

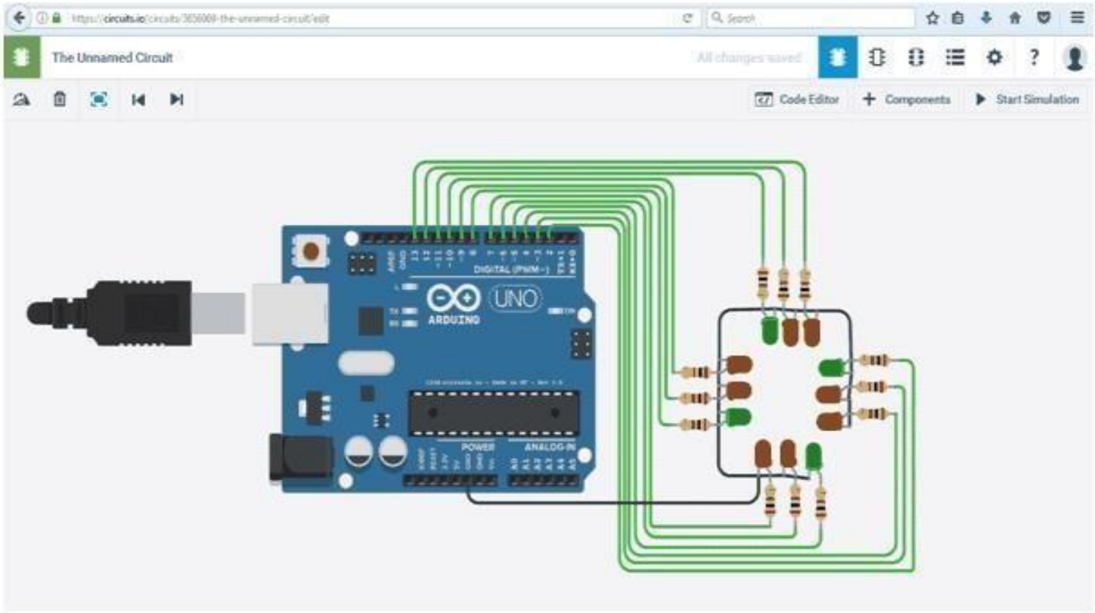
- Low cost, compact and powerful Wi-Fi Module
- Power Supply: +3.3V only
- Current Consumption: 100mA
- I/O Voltage: 3.6V (max)
- I/O source current: 12mA (max)
- Built-in low power 32-bit MCU @ 80MHz
- 512kB Flash Memory
- Can be used as Station or Access Point or both combined
- Supports Deep sleep (<10uA)
- Supports serial communication hence compatible with many development platform like Arduino
- Can be programmed using Arduino IDE or AT-commands or Lua Script

The ESP8266 is a low-cost Wi-Fi microchip, with built-in TCP/IP networking software, and microcontroller capability, produced by Espress if Systems in Shanghai, China. The chip was popularized in the English speaking maker community in August 2014 via the ESP-01 module, made by a third-party manufacturer Ai-Thinker.

User application: The ESP8266 Wi Fi Module is a self contained SOC with integrated TCP/IP protocol stack that can give any microcontroller access to your Wi Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor.

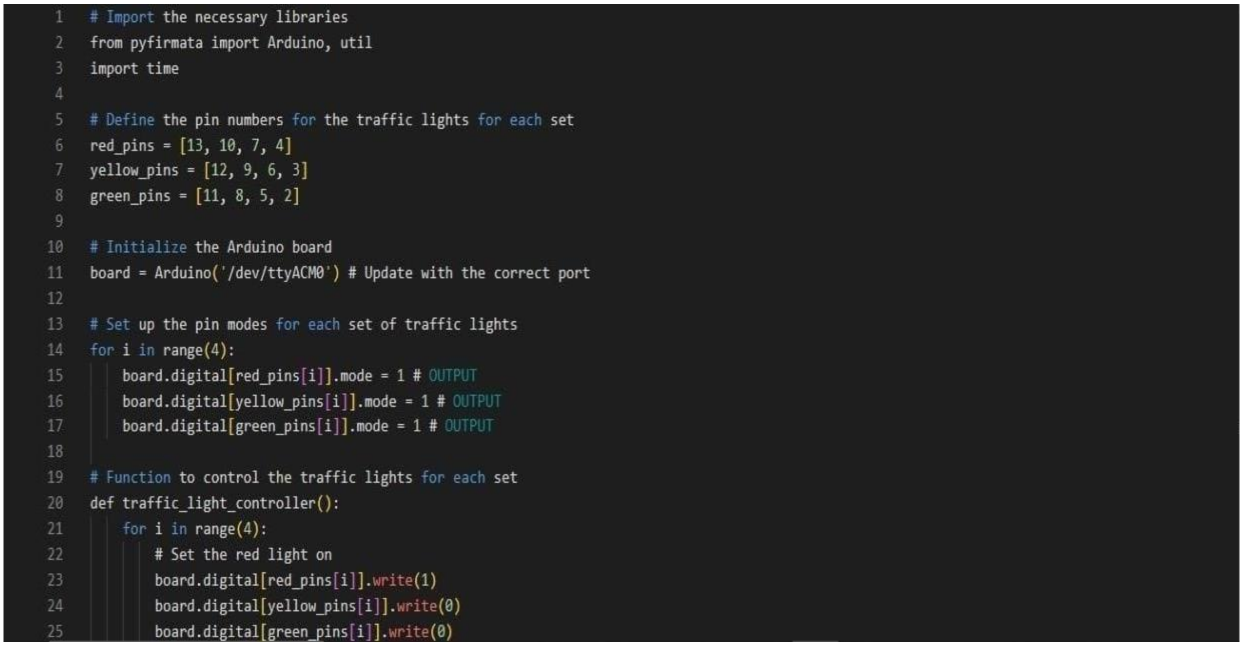
CIRCUIT DIAGRAM:

Connect All the LEDs as per the circuit diagram shown below for traffic light controller.

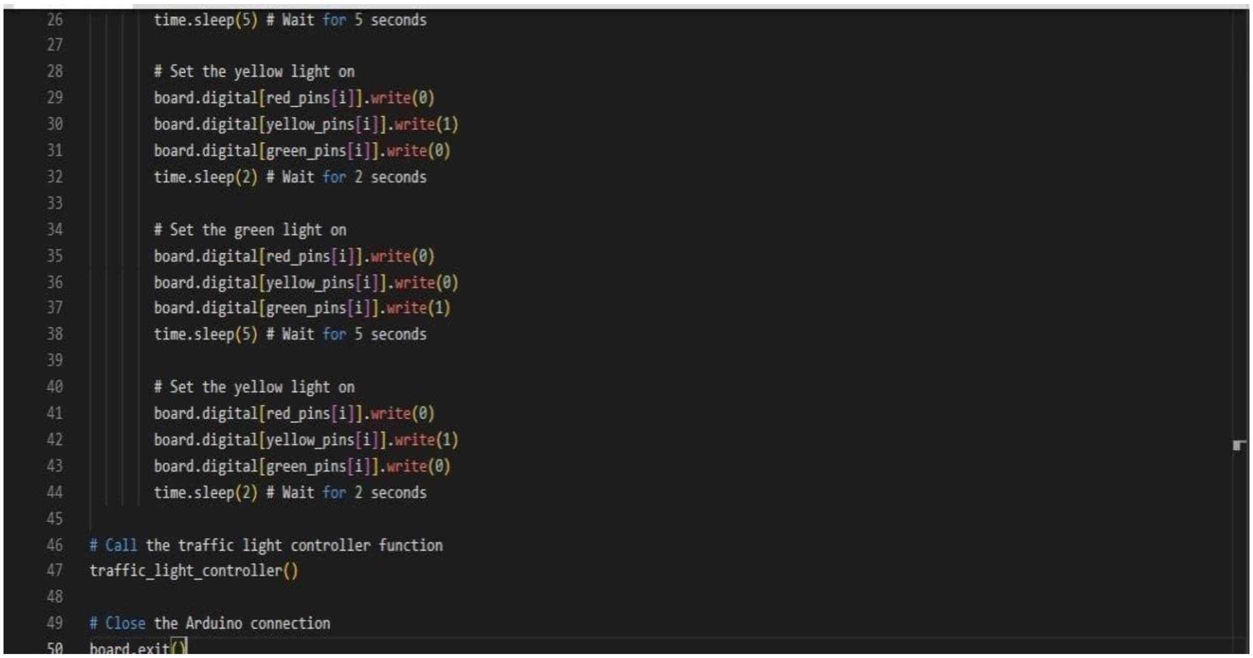


Note: In practical, we did not have to use the current limiting resistors as the current from each digital I/O pin of the Arduino UNO is limited to 20mA only. Hence no resistor required LED connection.

PROGRAM:

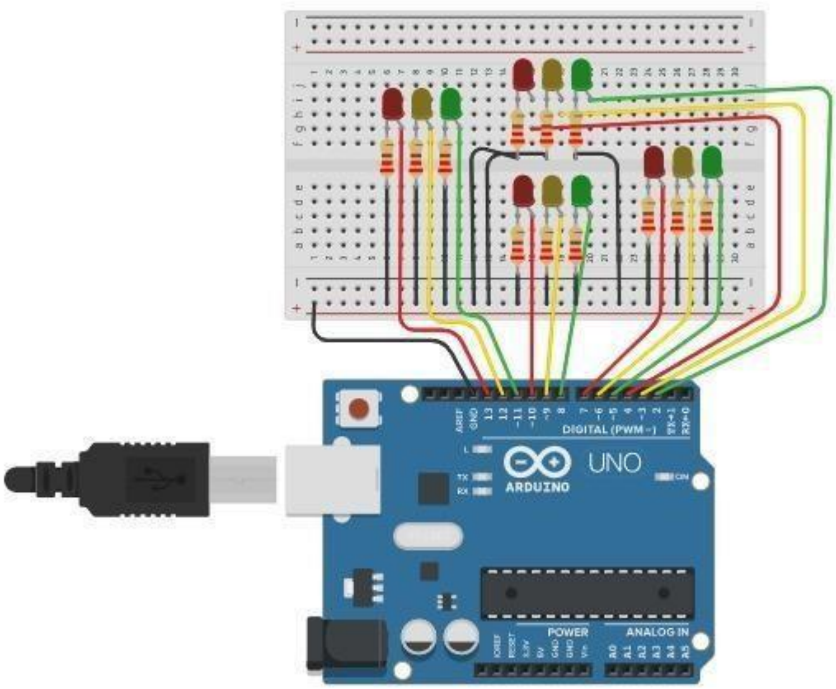


```
# Import the necessary libraries
from pyfirmata import Arduino, util
import time

# Define the pin numbers for the traffic lights for each set
red_pins = [13, 10, 7, 4]
yellow_pins = [12, 9, 6, 3]
green_pins = [11, 8, 5, 2]

# Initialize the Arduino board
board = Arduino('/dev/ttyACM0') # Update with the correct port

# Set up the pin modes for each set of traffic lights
for i in range(4):
    board.digital[red_pins[i]].mode = 1 # OUTPUT
    board.digital[yellow_pins[i]].mode = 1 # OUTPUT
    board.digital[green_pins[i]].mode = 1 # OUTPUT

# Function to control the traffic lights for each set
def traffic_light_controller():
    for i in range(4):
        # Set the red light on
        board.digital[red_pins[i]].write(1)
        board.digital[yellow_pins[i]].write(0)
        board.digital[green_pins[i]].write(0)
```



```
        time.sleep(5) # Wait for 5 seconds

        # Set the yellow light on
        board.digital[red_pins[i]].write(0)
        board.digital[yellow_pins[i]].write(1)
        board.digital[green_pins[i]].write(0)
        time.sleep(2) # Wait for 2 seconds

        # Set the green light on
        board.digital[red_pins[i]].write(0)
        board.digital[yellow_pins[i]].write(0)
        board.digital[green_pins[i]].write(1)
        time.sleep(5) # Wait for 5 seconds

        # Set the yellow light on
        board.digital[red_pins[i]].write(0)
        board.digital[yellow_pins[i]].write(1)
        board.digital[green_pins[i]].write(0)
        time.sleep(2) # Wait for 2 seconds

# Call the traffic light controller function
traffic_light_controller()

# Close the Arduino connection
board.exit()
```

OUTPUT:

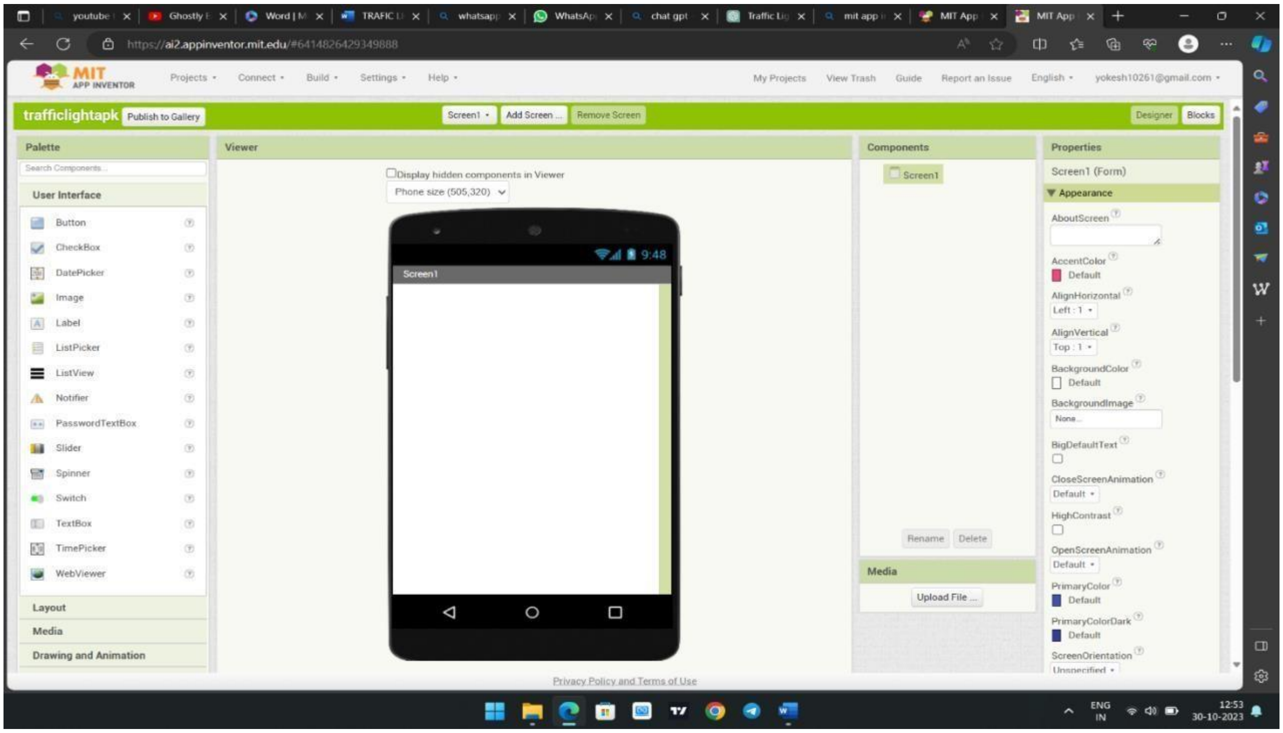


Why MIT app inventor?

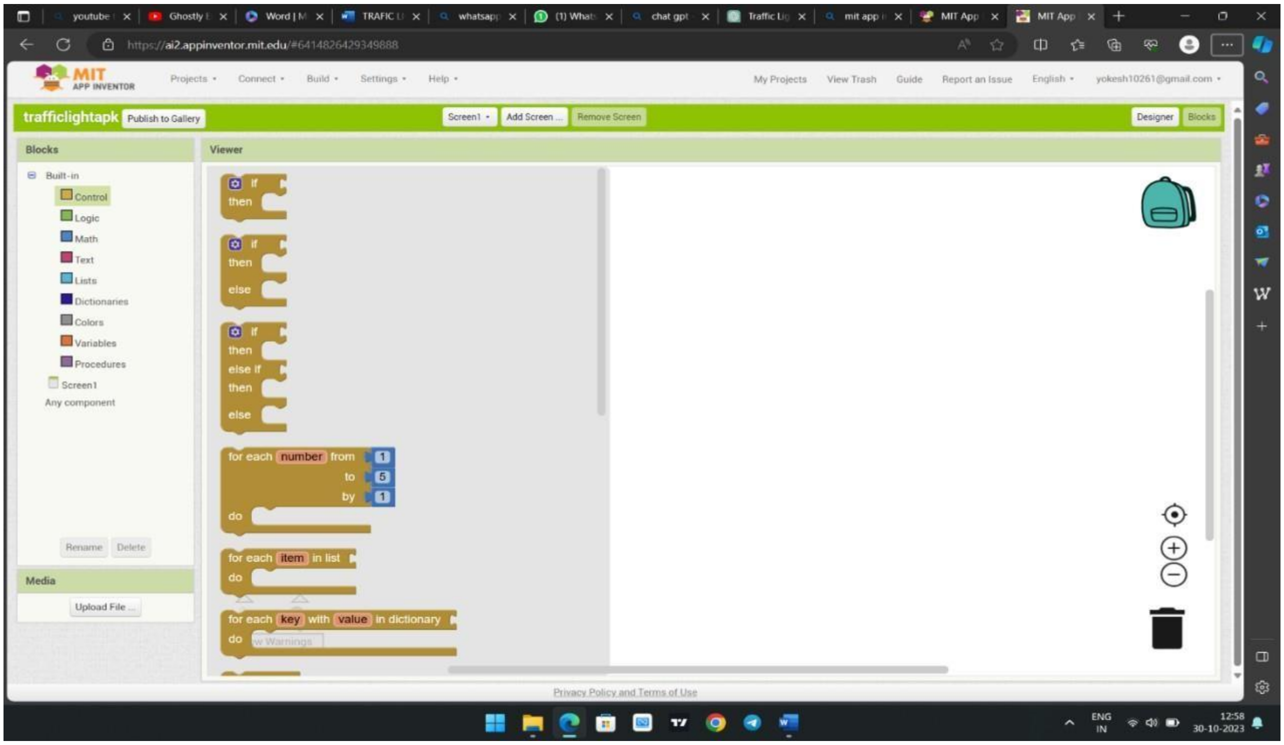
Traffic light application is very simple to use, and you can build any android application in minutes by just dragging and dropping components. Also, the MIT app inventor has been very popular among the young kids who start with STEM education, as block programming helps them understand the programming concept.

Introduction to MIT app inventor:

MIT app inventor window:



You can do the back-end development in the block menu, like how a specific button will behave when clicking on it.

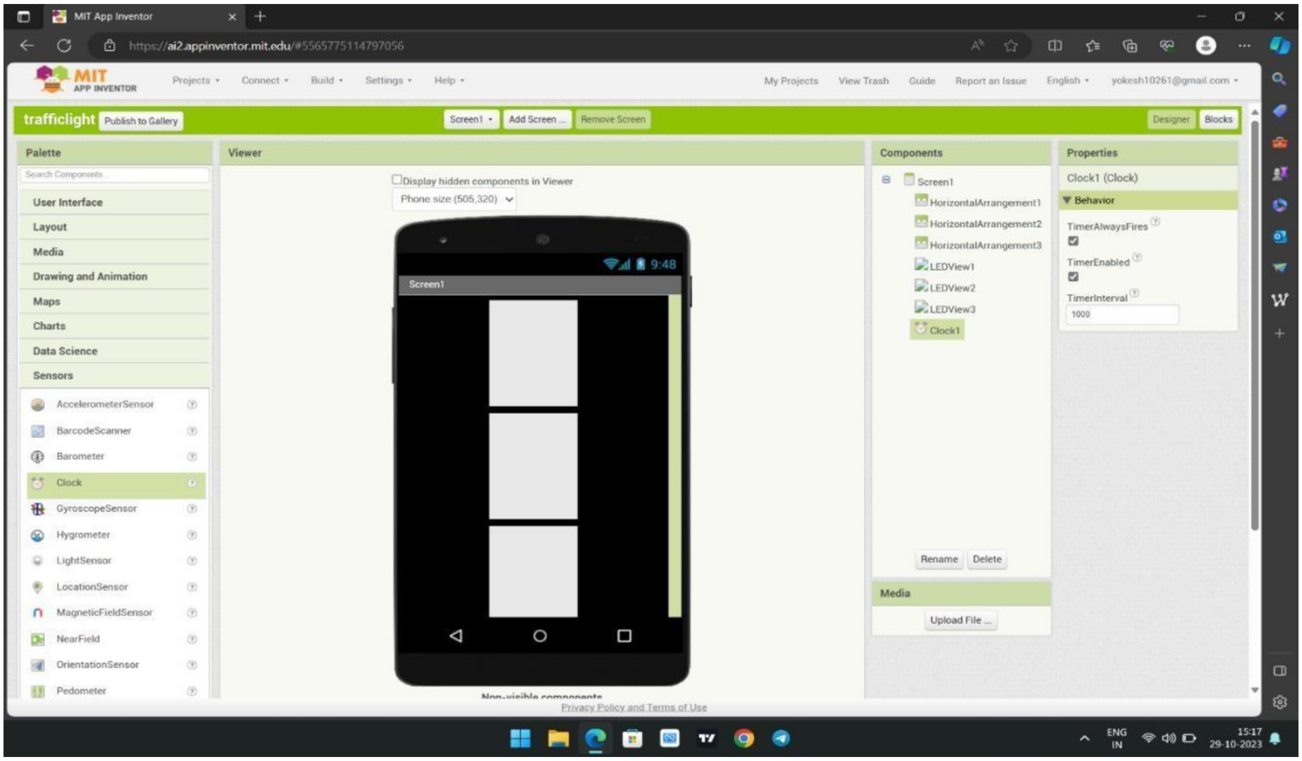


CREATING LAYOUT:

By dragging a vertical layout in our screen, all the items are in vertical arrangement and set the width of layout to fill-parent.

We will add a horizontal layout, and inside it, we will add a label to show the title TRAFFIC LIGHT dashboard, and in the right menu, you can play with parameters to adjust the size, width, height alignment, etc.

We need to add a button for connecting to the TRAFFIC LIGHT broker. So, I will copy the same horizontal layout, and instead of the image, I will press a button. Then rename it to “connect” and change the label text as well.



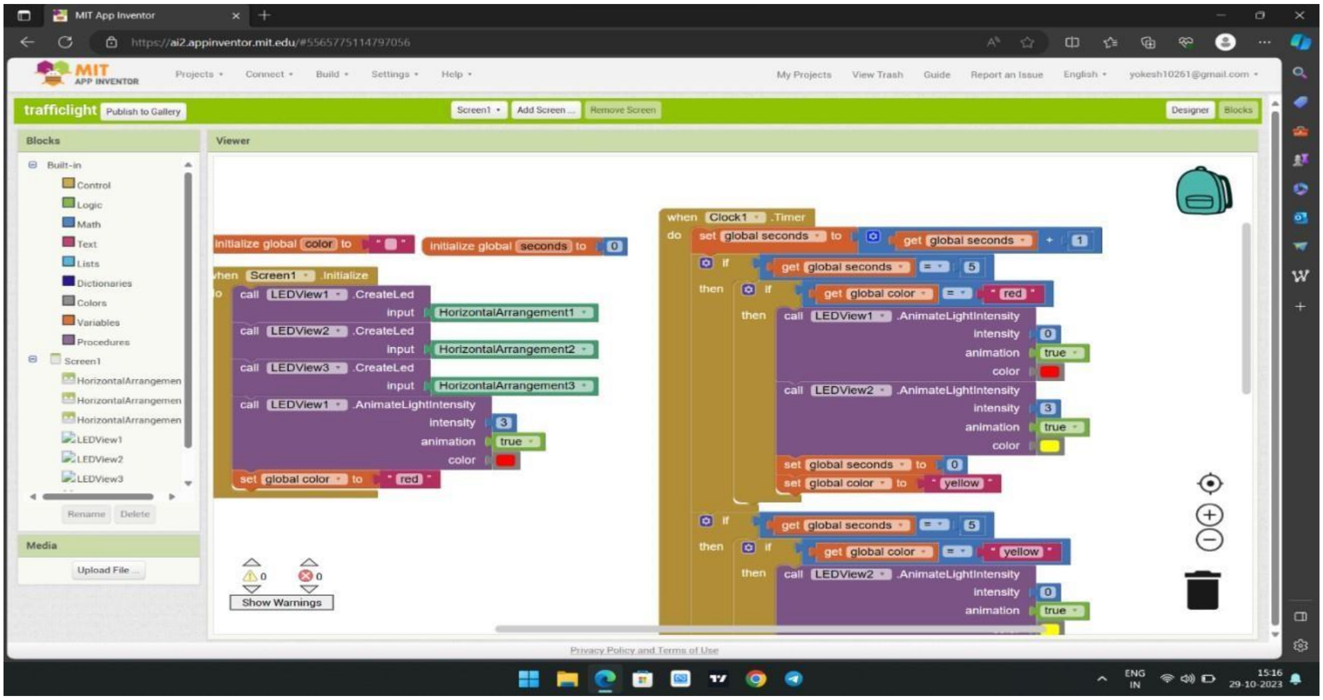
TimerAlwaysFires ☑

TimerEnabled ☑

TimerInterval 1000

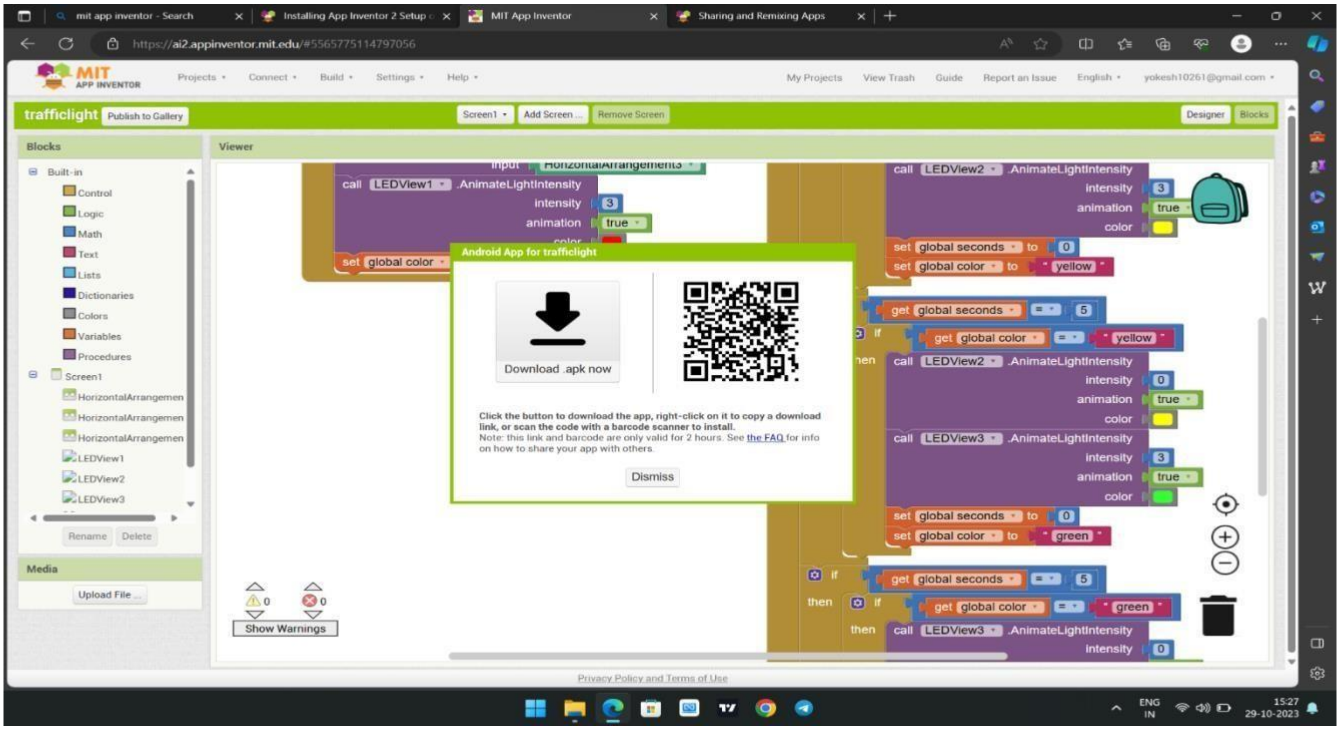
PROGRAMMING TRAFFIC LIGHT ANDROID APP:

we must download the LEDVIEW extension after the download the file will its l shown in the zip file extract the file after we upload the file to the Mit app inventor, and it will show like



```
initialize global color to " "
initialize global seconds to 0

when Screen1 .Initialize
do  call LEDView1 .CreateLed
         input HorizontalArrangement1
    call LEDView2 .CreateLed
         input HorizontalArrangement2
    call LEDView3 .CreateLed
         input HorizontalArrangement3
    call LEDView1 .AnimateLightIntensity
         intensity 3
         animation true
         color
    set global color to " red "

when Clock1 .Timer
do  set global seconds to get global seconds + 1
    if get global seconds = 5
    then if get global color = " red "
         then call LEDView1 .AnimateLightIntensity
                   intensity 0
                   animation true
                   color
              call LEDView2 .AnimateLightIntensity
                   intensity 3
                   animation true
                   color
              set global seconds to 0
              set global color to " yellow "
    if get global seconds = 5
    then if get global color = " yellow "
         then call LEDView2 .AnimateLightIntensity
                   intensity 0
                   animation true
```

Building APK and running it:



Android App for trafficlight

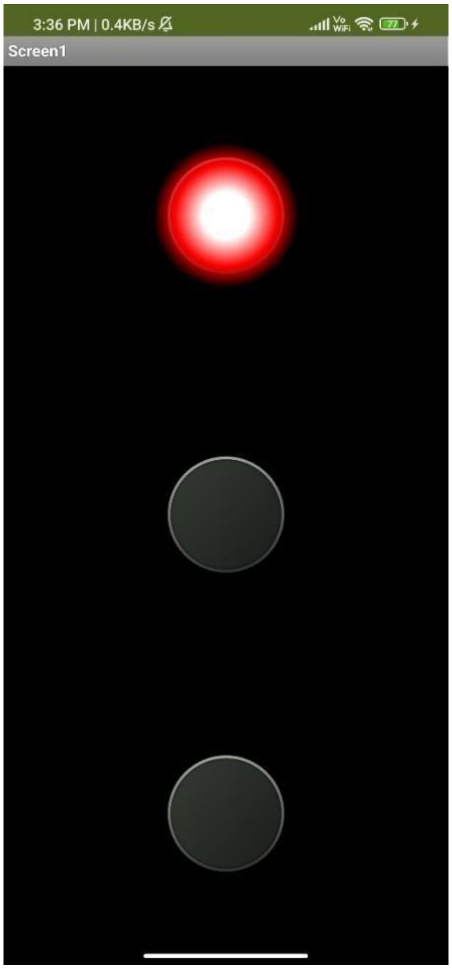
Download .apk now

Click the button to download the app, right-click on it to copy a download link, or scan the code with a barcode scanner to install.
Note: this link and barcode are only valid for 2 hours. See the FAQ for info on how to share your app with others.

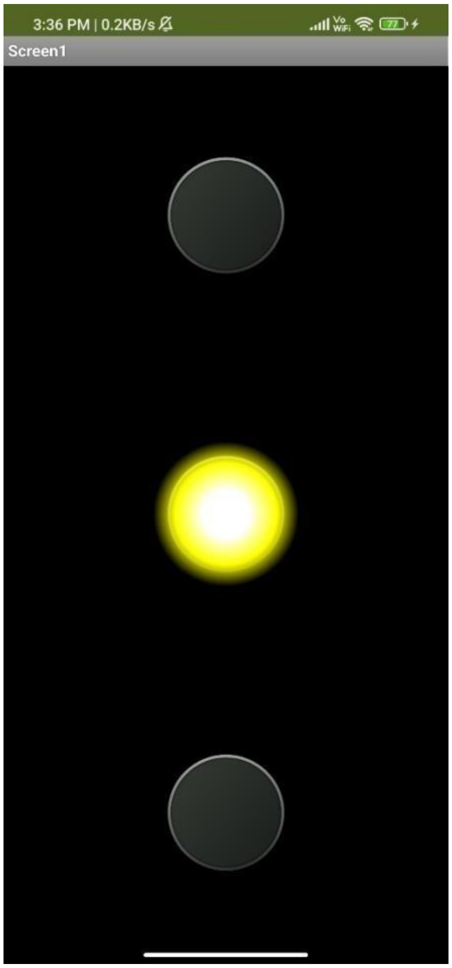
Dismiss

OUTPUT:

Click on build and then android app (Apk), you will get two options either to download it or you can directly download from your android device by scanning the code



- The Traffic light application blows RED Light in Five seconds.

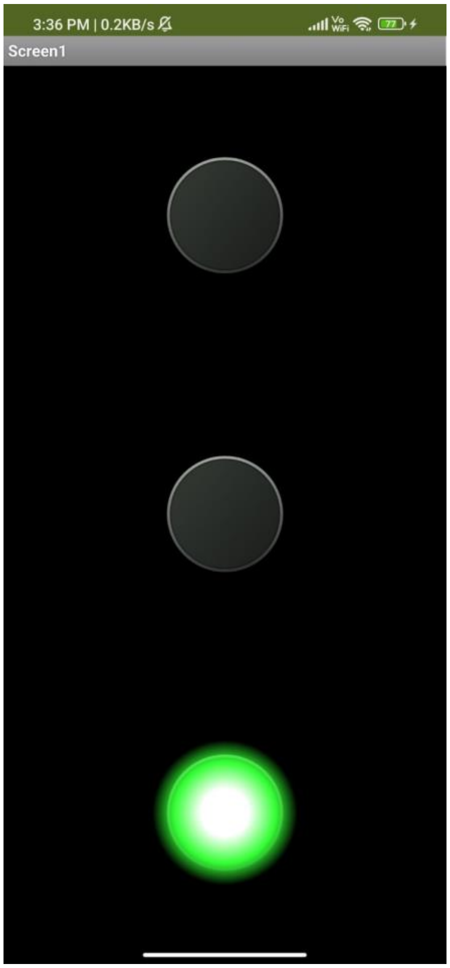


- After five seconds, it will blow YELLOW light.

- After the yellow light it will blow GREEN light.

APP LOGIC:

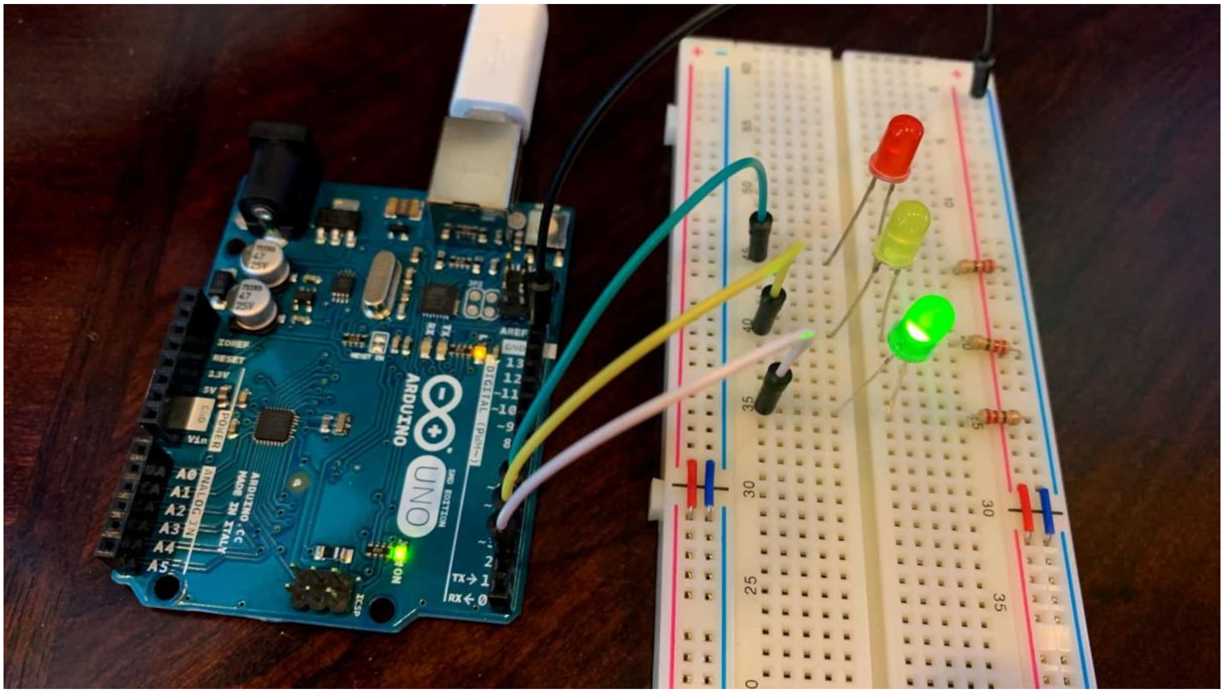
The app would have underlying logic to control the behavior of the traffic light. This logic determines when each light should be turned on and for how long. Typically, a traffic light follows a sequence



- Red light: Stop

- Yellow light: Prepare to move
- Green light: Go

To see my project using MIT app inventor I attached my app below:


ARDUINO
UNO
DIGITAL (PWM~)
AREF
GND
13
12
~11
~10
~9
8
TX→1
RX←0
ON
ICSP
IOREF
RESET
3.3V
5V
GND
Vin
POWER
ANALOG IN
A0
A1
A2
A3
A4
A5
ARDUINO.CC
MADE IN ITALY
45
40
35
30
25
30
35

CONCLUSION:

- There are several traffic light apps available on the Google Play Store.

One such app is My Traffic Light .

- This app lets you control the order of signal changes via preset cycles and

lets you change the signal light cover patterns by choosing one of three

lenses (a blank lens, a striped lens, or an LED lens) .

- You can use the automated timer or turn it off and control when the

signal changes manually .

- Another app is Traffic light detection & MP3 . This app can automatically

lock the red light or manually lock the red or green light, and notify the

user immediately if the light changes .

- When you wait for the traffic lights, you don’t have to keep looking at the

lights .