PHASE 3:

DEVELOPMENT PART 1:

TRAFFIC MANAGEMENT

SYSTEM

The development of a traffic management system involves several stages, and here is an overview of the first part:

1. Project Planning and Requirements Gathering: In this initial phase, the project team defines the scope and objectives of the traffic management system.

2. Sensor Deployment: The next step is to deploy the sensors in strategic locations throughout the city. These sensors can include traffic cameras, vehicle detectors, and environmental sensors to monitor air quality. The sensors are connected to the IoT network, allowing them to transmit data in real-time.

3. Network Infrastructure Setup: A reliable network infrastructure is crucial for connecting the IoT devices.

4. Data Collection and Processing: Once the sensors are installed and connected, they start collecting data on traffic conditions. This data includes vehicle counts, speeds, and other relevant parameters.

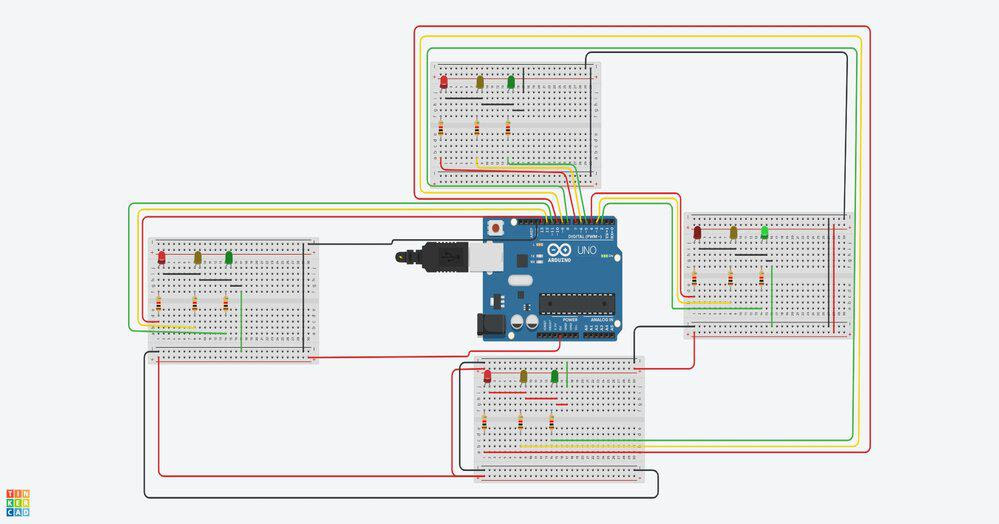
This is the initial part of the development process for a traffic management system in the context of IoT.

IOT BASED TRAFFIC MANAGEMENT SYSTEM USING ARDUINO:

**Required components:**

1. Arduino
2. 12 pieces of 220-ohm resistors
3. 4 pieces of breadboard
4. Connecting wires
5. Red, yellow, and green LEDs

## **Circuit Diagram:**



Arduino is an open-source electronics platform that is commonly used for building various electronic projects. It consists of a microcontroller board with digital and analog input/output pins, which allows it to be programmed and connected to different electronic components.

The Arduino board acts as the brain of the project, receiving input from sensors and performing certain actions based on the program or code uploaded to it. It can interact with various components such as LEDs, motors, sensors, and more.

In this specific project, you would require 12 pieces of 220-ohm resistors. Resistors are used to limit the flow of current and protect the components from damage. The 220-ohm resistors are typically used with LEDs to ensure they receive the correct amount of current.

You would also need 4 pieces of breadboard, which are modular boards with interconnected holes that allow you to easily connect and prototype electronic circuits. Breadboards provide a convenient way to test and build circuits without having to solder them together.

Connecting wires are essential for establishing connections between components on the breadboard. These wires connect the Arduino board, resistors, and LEDs, allowing them to communicate and function together.

Lastly, you would need red, yellow, and green LEDs. LEDs (Light Emitting Diodes) are electronic components that emit light when current flows through them. Red, yellow, and green LEDs have different colors and can be used for various purposes in electronic projects. For example, red LEDs can be used as indicators for errors or warnings, while green LEDs can indicate a successful operation or good status.

To summarize, the Arduino is a versatile platform that allows you to create and control electronic projects. In this specific project, you would use resistors, breadboards, connecting wires, and LEDs to build a circuit that interacts with the Arduino board. The resistors, breadboards, and wires help establish the necessary connections, while the LEDs provide visual feedback or indications.

PYTHON PROGRAM USING ARDUINO:

1-Assign the traffic lights pins to variables

int d\_red =10;

int d\_yellow =9;

int d\_green =8;

int r\_red =4;

int r\_yellow =3;

int r\_green =2;

int l\_red =13;

int l\_yellow =12;

int l\_green =11;

int u\_red =7;

int u\_yellow =6;

int u\_green =5;

2-Configure the traffic lights as outputs

void setup()

{

pinMode(d\_red, OUTPUT);

pinMode(d\_yellow, OUTPUT);

pinMode(d\_green, OUTPUT);

pinMode(r\_red, OUTPUT);

pinMode(r\_yellow, OUTPUT);

pinMode(r\_green, OUTPUT);

pinMode(l\_red, OUTPUT);

pinMode(l\_yellow, OUTPUT);

pinMode(l\_green, OUTPUT);

pinMode(u\_red, OUTPUT);

pinMode(u\_yellow, OUTPUT);

pinMode(u\_green, OUTPUT);

}

3-Use **loop** function to keep the lights in a loop and use **changeLIght()** function to carry out the logic

void loop()

{

changeLights();

}

void changeLights()

{

//Start (all yellow)

digitalWrite(u\_red,LOW);

digitalWrite(d\_red,LOW);

digitalWrite(r\_red,LOW);

digitalWrite(l\_green,LOW);

digitalWrite(u\_yellow,HIGH);

digitalWrite(d\_yellow,HIGH);

digitalWrite(r\_yellow,HIGH);

digitalWrite(l\_yellow,HIGH);

delay(5000);

//upper lane go

digitalWrite(u\_yellow,LOW);

digitalWrite(d\_yellow,LOW);

digitalWrite(r\_yellow,LOW);

digitalWrite(l\_yellow,LOW);

digitalWrite(u\_green,HIGH);

digitalWrite(r\_red,HIGH);

digitalWrite(l\_red,HIGH);

digitalWrite(d\_red,HIGH);

delay(10000);

//ALL YELLOW

digitalWrite(u\_yellow,HIGH);

digitalWrite(d\_yellow,HIGH);

digitalWrite(r\_yellow,HIGH);

digitalWrite(l\_yellow,HIGH);

digitalWrite(u\_green,LOW);

digitalWrite(r\_red,LOW);

digitalWrite(l\_red,LOW);

digitalWrite(d\_red,LOW);

delay(5000);

//RIGHT LANE GO

digitalWrite(u\_yellow,LOW);

digitalWrite(d\_yellow,LOW);

digitalWrite(r\_yellow,LOW);

digitalWrite(l\_yellow,LOW);

digitalWrite(u\_red,HIGH);

digitalWrite(l\_red,HIGH);

digitalWrite(d\_red,HIGH);

digitalWrite(r\_green,HIGH);

delay(10000);

//ALL YELLOW ON

digitalWrite(u\_yellow,HIGH);

digitalWrite(d\_yellow,HIGH);

digitalWrite(r\_yellow,HIGH);

digitalWrite(l\_yellow,HIGH);

digitalWrite(u\_red,LOW);

digitalWrite(l\_red,LOW);

digitalWrite(d\_red,LOW);

digitalWrite(r\_green,LOW);

delay(5000);

//DOWN LANE GO

digitalWrite(u\_yellow,LOW);

digitalWrite(d\_yellow,LOW);

digitalWrite(r\_yellow,LOW);

digitalWrite(l\_yellow,LOW);

digitalWrite(u\_red,HIGH);

digitalWrite(l\_red,HIGH);

digitalWrite(r\_red,HIGH);

digitalWrite(d\_green,HIGH);

delay(10000);

//ALL YELLOW

digitalWrite(u\_yellow,HIGH);

digitalWrite(d\_yellow,HIGH);

digitalWrite(r\_yellow,HIGH);

digitalWrite(l\_yellow,HIGH);

digitalWrite(u\_red,LOW);

digitalWrite(l\_red,LOW);

digitalWrite(r\_red,LOW);

digitalWrite(d\_green,LOW);

delay(5000);

//LEFT LANE GO

digitalWrite(u\_yellow,LOW);

digitalWrite(d\_yellow,LOW);

digitalWrite(r\_yellow,LOW);

digitalWrite(l\_yellow,LOW);

digitalWrite(u\_red,HIGH);

digitalWrite(d\_red,HIGH);

digitalWrite(r\_red,HIGH);

digitalWrite(l\_green,HIGH);

delay(10000);

}

HOW TO WORKING IS PROGRAM:

This program uses the Arduino code to control the traffic lights. It starts with all lights in the "yellow" state, then changes to the "upper lane go" state, followed by "all yellow", "right lane go", "all yellow", "down lane go", "all yellow", and finally "left lane go".

Each state is controlled by setting specific pins HIGH or LOW to turn on or off the corresponding LEDs. The delay() function is used to pause the program for a specific amount of time between each state change.

By continuously looping the changeLights() function in the loop() function, the traffic lights will keep changing in a loop and simulate a real traffic light system.

OUTPUT:

The output of this program is a simulation of a traffic light system. The traffic lights will cycle through different states according to the code logic.

Here is the sequence of the traffic light states:

1. Start (all yellow): All lights are yellow.

2. Upper lane go: Upper lane lights are green, while the rest are red.

3. All yellow: All lights are yellow.

4. Right lane go: Right lane lights are green, while the rest are red.

5. All yellow: All lights are yellow.

6. Down lane go: Down lane lights are green, while the rest are red.

7. All yellow: All lights are yellow.

8. Left lane go: Left lane lights are green, while the rest are red.

The program will keep cycling through these states indefinitely in a loop. Each state lasts for a specified duration controlled by the delay() function in the code.

THANK YOU