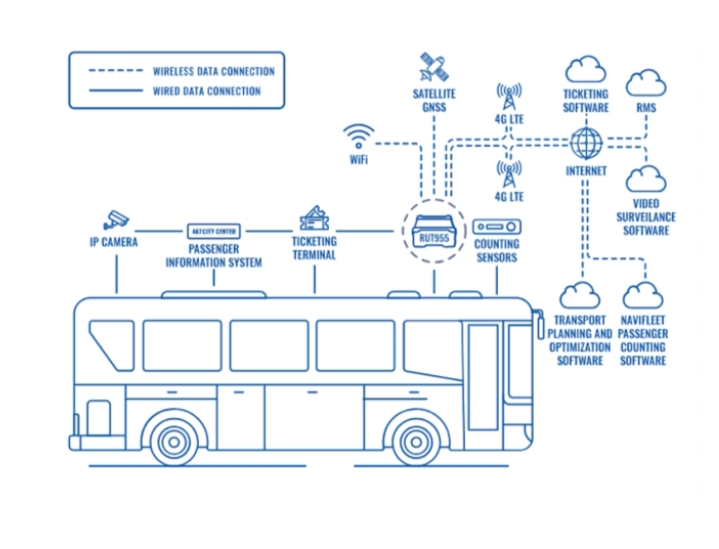
1. **NSN COLLEGE OF ENGINEERING AND TECHNOLOGY-GROUP 4**

**PHASE-3 PUBLIC TRANSPORTATION AND OPTIMIZATION**

**TEAM MEMBERS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **NAME** | **DEPRT** | **MENTOR NAME** |
| 1 | Janakidevi R | CSE | MRS.D.Shanthipriya AP/CSE |
| 2 | Ranjitha M | CSE |
| 3 | Thavasimani S | CSE |
| 4 | Rejiya Sulthana N | CSE |
| 5 | Astalaksmi S M | CSE |

**BLOCK DIAGRAM**

****

**Passenger Information System :**

It includes information management on routes and their display via different display elements deployed at passenger platforms. These devices may be of different types:

* Screens
* Indicators at platforms
* Indicator panels in general

The layout of all information to be displayed may be adapted to the needs of different clients and languages.

Furthermore, the same visual and audio means can be used to display corporate or commercial information at passenger platforms.

**Counting Sensors :**

Implementing accurate and reliable passenger detection and counting system is an important task for the correct distribution of available transport system. The aim of this paper is to develop an accurate computer vision-based system to track and count passengers. The proposed passenger detection system incorporates the ideas of well-established detection techniques and is optimally customised for both indoor and outdoor scenarios. The candidate foreground regions (inside an image) are extracted in the proposed method and are described using the histograms of oriented gradient descriptor. These features are trained and tested using support vector machine classifier and the detected passengers are tracked using a filter. The proposed counting system is used to count passengers automatically when they pass through a virtual line of interest. Accuracies ranging 91.2 percent to 86.24 percent were found for passenger detection using the proposed passenger detection and counting system whereas relative counting errors varied ten percent to thirteen percent.

**Ticketing Counting :**

The Bus Terminal Ticketing System. In the research, passengers are divided into post-purchase and direct purchase according to the characteristics of fare- paying, and the day is divided into three periods according to the average arrival time step. In the three time periods, simulation software AutoMod is applied to simulate and run the queuing system in order to optimize the number of ticket window sales and improve the efficiency of the system.

**Transportation planning and optimization software :**

First of all, transportation planning software (TPS) is similar to transportation management systems (TMS). Both TPS and TMS software help logistics businesses with daily tasks that include:

* Estimating rates
* Load planning and routing
* Tracking carriers
* Shipment tracking
* Data management
* Customer management
* Accounting and audits
* Tracking key performance indicators (KPIs)
* Integrating with other software platforms and providers

**IP Camera :**

Our report on the Global Panoramic IP Cameras Market is an extensive, comprehensive analysis of the current market trends and drivers. It provides an in-depth understanding of the opportunities in the industry, helping businesses make informed decisions and capitalize on the rapidly growing market.

import random

import time

# Simulating IoT data collection from buses

class Bus:

def \_init\_(self, bus\_id, route\_id, current\_location, speed):

self.bus\_id = bus\_id

self.route\_id = route\_id

self.current\_location = current\_location

self.speed = speed

def update\_location(self):

# Simulating random movement

self.current\_location += self.speed \* random.uniform(0.8, 1.2)

def get\_location(self):

return self.current\_location

# Simulating data processing and optimization

class TransportOptimizer:

def \_init\_(self, buses):

self.buses = buses

def optimize(self):

for bus in self.buses:

bus.update\_location()

def get\_bus\_locations(self):

return {bus.bus\_id: bus.get\_location() for bus in self.buses}

# Simulating the execution of the script

if \_name\_ == '\_main\_':

buses = [Bus(1, 'A', 0, 30), Bus(2, 'B', 10, 25)] # Simulated buses with initial parameters

optimizer = TransportOptimizer(buses)

for \_ in range(5): # Simulating 5 iterations

optimizer.optimize()

print(optimizer.get\_bus\_locations())

time.sleep(2) # Simulating time delay between iterations