**Assignment – 1**

**Smart Factory Ventilation System**

**Introduction**

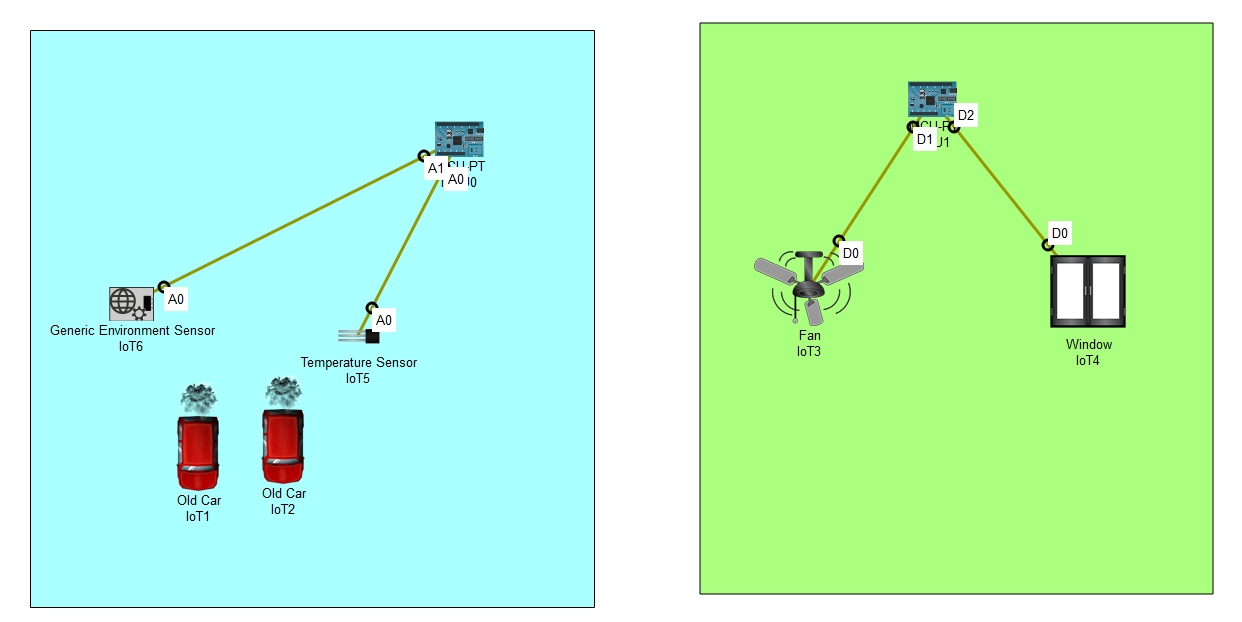
The ventilation system is very important for any house or factory as it helps us breathe without suffocation. This report represents a smart factory, in which ventilation can be controlled automatically. For instance, if there is any temperature rise from a threshold value, the fan will be turned on. And, if there is a sudden increase in CO level, the window will be opened. And, this whole system is controlled using 2 MCUs, a temperature sensor, a CO sensor, and an external server (Python). This implementation explains how we can visualize data in real-time case along with graphical animations.

**Overview**

The platforms that are mainly used here are – Packet Tracer (to give a visual representation of how it works) and Python (to control the system from outside). This design includes a few components to visualize data and act accordingly. Such as-

* MCU1: It is mainly used to collect data from sensors which data vary according to the parameters of the environment.
* MCU2: It is mainly used to collect sensor data from the server and acts accordingly to give instructions to actuators like fans, motors, or any other components.
* Cloud: It works as the gateway between MCU1 and MCU2 by using a TCP connection between them. It reads data from MCU1 and pushes it to MCU2. For this project, I have used Python to act like a server or cloud. Moreover, the data are plotted using matplotlib.
* Sensor: Mainly, 2 types of sensors are used here – temperature and CO sensor.

Here, it a graphical view of implemented system-



Temperature sensor Threshold: 38

CO (generic environment sensor) Sensor Threshold: 20

**Mechanism**

1. The server waits for MCU1 to build a TCP connection with it.
2. Upon connecting with the MCU1, server again waits for MCU2 to build the TCP connection.
3. Then it starts reading data periodically from MCU1 and sends it to MCU2.
4. MCU2 processes the data and commands the fan to rotate and windows to open or close accordingly. When the temperature exceeds the threshold value fan will be turned on and vice-versa. When the temperature is lower than the threshold value, a window will be opened and vice-versa.
5. On the other hand, matplotlib continuously plots the graph of temperature and CO level along with fan and window status.

**Protocol Messages**

* MCU1 to server: format – (sending, temperature, CO level)

A white rectangular object with a gray border

AI-generated content may be incorrect.

* Received message in server from MCU1: format - (Received from MCU1: temperature, CO level)

A black background with white text

AI-generated content may be incorrect.

* Server to MCU2: format – (Sent to MCU2: fan\_status;window\_status)

A black background with white text

AI-generated content may be incorrect.

* Received message in MCU2: format – (received: fan\_status;window\_status)



* MCU2 to server: format - MCU2 Response: Fan turned Fan\_statusWINDOW is window\_status



**Challenges and Solution**

* **Temperature variation**: It’s very critical to handle temperature variation. First, it didn’t show any variation even when I changed the environmental parameter or used old car. That’s why I had to use *random.uniform* function to see the fluctuation of temperature.
* **Temperature changes with time**: Again, it’s hard to control. Because the temperature sensor started to show different values and ranges throughout the day. For example, in the daytime, I noticed temperatures fluctuated between 20-35 degrees but at night (at the time of writing this report) it fluctuated between 35-48 degrees. That’s why I had to change the threshold value from 28 to 38 while simulating it at late night.
* **CO sensor**: CO sensor in the packet sensor doesn’t work so I tried to use a generic environmental sensor, but it didn’t show any data even changing environmental parameters or using old car. Sadly, I couldn’t make this part. That’s why I tried to control the window using the temperature value.

**Results Analysis**

* Successful TCP connection built between MCUs and Server
* Temperature variation throughout the day
* FAN and Window commands work with temperature variation
* Graphical visualization of data using matplotlib

Here, is the graphical representation from matplotlib-

