General Questionnaire:

1. A hyperloop in India can transform transportation by solving problems such as overcrowding, long travel times, and high emissions. It can be fast, sustainable, and efficient, with the potential to connect cities like Chennai and Bengaluru in under 30 minutes, thereby enhancing economic growth and quality of life. Of course, challenges such as cost and land acquisition exist, but technology is advancing, and innovative teams like Avishkar Hyperloop make it possible. Even Though it might not look like, but this concept is a very feasible idea and could be achieved in the mere future
2. Ever since I entered IITM the very first place that I saw wasn't my hostel but rather the CFI building and specifically the first team that I saw was the Avishkar team (during the demo day 2024). I wasn't sure what you guys were doing and then through some of my friends I found out about your mission and this was something that motivated me. I for a fact knew that creating a pod to transport people was no easy task but you guys managed to pull it. This was also something which peaked my interest to join the team. I have quite some experience in python and C. So I think that with my creative thinking and coding skill could be of help to the team in some way and I could assure you that ***if there is a problem given to me I could find out a solution fast and an efficient one too*** .I think that this would be pretty helpful for creating a good resume in cp which is something that i'm pretty much interested in (I have attended a lot of sessions conducted by the programming club).
3. I know that cp is the one which would interest me a lot and thus I would like to work on the GUI part of the pod. A lot of people don't understand how important the GUI subsystem is. It is the only way in which the engineer and the person using the pod could communicate. So it must be perfected to its finest form and I hope that I could be a part of this. The part that interests me is that the GUI must always be reloaded ie. it must have a refresh rate in terms of milliseconds. This is something that also interested me into the GUI subsystem. I would specifically like to work on the team where I could implement machine learning to predict the errors before they appear so that the person inside the pod is left undisturbed.

4. The **TUM hyperloop** developed custom battery cells with high power density and improved power electronics, contributing to a record speed of 482 km/h.The **Delft hyperloop** developed the **HELIOS III** pod which introduced the first fully scalable lane-switching mechanism, bringing the concept closer to a full-scale hyperloop network. The **HyperXite** developed a fail-scale friction based braking system enhancing safety by providing a reliable stopping mechanism.
5. I am currently not enrolled in any POR this year. Hopefully this could be my first POR 🙂. I could dedicate an average time of about 5 hours/day(not lying). Cause i'm pretty much free for the entire day except for the class timing.
6. Here is a list of software that I am familiar with:
    - Python(coding for about 3 years)
    - C(coding for about a year)
    - Html(learning for about a year)
    - CSS(learning for about a year)
    - AutoCAD(beginner)
    - js(beginner)
    - Linux (bash)(intermediate)

      Not too sure if I could include these too

    - Neovim
    - Sublime text
    - Burpsuite

Question No. 2:

1.What is MQTT and how does it work?

- MQTT(Message Queuing Telemetry Transport) is a lightweight messaging software which is mainly used in devices which have very limited resources and unreliable network connections. It is very simple and is also widely used because of its efficiency.
- **How does it work?** The key components in MQTT are **Broker, Publisher, Subscriber**. A client would have subscribed to a particular **channel** (which would be present in something like a directory in linux folders i.e. /user/Desktop/topic or something like that). After the subscription of the client to the particular channel he becomes a **subscriber**. There would be another client who could upload payloads to the specific

topics known as a **publisher.** The **broker** is the one who manages the message distribution. It ensures that the message sent by the publisher to the specific topic is reached to all the subscribers. Now the question is how does the broker and the subscriber connect with each other. They do so over the TCP or IP protocols.

- The MQTT is used in smart homes where the smart appliances are connected to the broker(in some cases it is the phone). And the publisher may be the sensors which send the messages to the broker and the broker sends the messages to the subscriber ie. the smart appliances.
- **How does MQTT ensure lightweight and efficient messaging?** The MQTT makes use of the minimal header packet transmission ie. the header packet is only up to 2 bytes thus reducing the network usage by a significant amount. Another interesting mechanism is used in MQTT which ensures that the connection is still “alive”. It does this by the following way:
    - The client sends a pingreq packet which the broker then receives and sends a pinresp packet.
    - These occur in a small time interval to ensure that the connection is still alive.
    - If the client does not send any pingreq packet then the connection is lost and it tries to reconnect.
- There is also something called quality of service (QOS) which ensures that the message is delivered according to the application’s need. There are three modes such as **QOS0,QOS1,QOS2.**
- The **QOS0** ensures that the message sent is not retransmitted so that there are no duplicates formed with the best delivery effort.
- The **QOS1** ensures that the message is delivered but there is a duplication which could be formed. So it isn’t very secure.
- The **QOS2** ensures that the message is delivered and there are no duplicates that are formed.

2.

1.How can MQTT be used to get real time hyperloop pod data?

- We must have a clear plan to ensure that all parts work in unison. The Publishers in the system can be thought of

as the sensors placed inside the hyperloop pod. These sensors are very important because they collect significant data, like speed, pressure, and temperature. For example, the data streams may be named as /hyperloop/speed, /hyperloop/pressure, and /hyperloop/temperature.

- The broker's role is also not irrelevant. This can be performed by options such as MOSQUITTO or HIVEMQ. Although this decision is not critical in nature, it does not matter much because they both perform the same function without fail. They allow proper communication through the management of messaging between Publishers and Subscribers.
- The subscribers in this configuration are the devices that will receive the data. They will capture information from the specified topics, including speed, pressure, temperature, and possibly other relevant topics that may come up.
- Using MQTT for message transmission is one way through which this process will occur. One of the benefits of MQTT is its QOS security service. The essence of this security service is to ensure that messages are safely received without duplication, even if the network connection is unstable.
- Furthermore, MQTT is lightweight and efficient. These reasons have been critical in using it within hyperloop pods. It is designed to transmit data quickly while ensuring as little resource use as possible. All these features make MQTT suitable for the demanding environment of travel in hyperloops.

2. Difference between the MQTT and HTTP for data communication.

- The difference between MQTT and HTTP is that communication in MQTT is mainly between the broker and the subscribers and between the broker and the publishers. On the other hand, HTTP enables direct communication between the client and the server. This difference in kind of interactions points to different types of interactions these protocols support.
- Now, regarding the packet headers, the MQTT has a 2 byte compact header packet and this is for efficient data

transfer. On the other hand, HTTP has larger header packets in order to effectively manage communication between the server and client. This difference in header sizes is a good example of how two protocols can have entirely different approaches to data exchange.

- MQTT also has Quality of Service (QoS) mechanisms that improve security making it suitable for secure data transfer. On the other hand, HTTP relies solely on TCP for its communication and this makes it entirely dependent on the existing network protocols for data transfer.
- In practical use, MQTT is well suited for real time, low power consumption networks and so is a good choice for monitoring of IoT devices. The lightweight design and the emphasis on resource efficiency makes it suitable for such situations. On the other hand, HTTP is more useful in web applications that do not require real time data transfer. This difference in the application of these protocols reveals their ability to be used in different scenarios and environments.

3.

1.Eclipse Mosquitto MQTT Broker

Eclipse Mosquitto is a lightweight, open-source message broker that implements the MQTT protocol. It is ideal for IOT devices and devices with minimal bandwidth usage. It’s available for linux, windows, macos and many other platforms. It also supports TLS encryption for secure communication. It could be used for real time messaging in IOT devices and communication between sensors and actuators.

2.HiveMQ MQTT Essentials

It is a series of guides designed for understanding the MQTT protocol.

4.

```
import paho.mqtt.client as m
import random
import time
broker = "localhost"
port = 1883
topic = "hyperloop/data"

client = m.Client() # used for creating an instance of an mqtt client
client.connect(broker,port)# the newly created instance of the client is used to connect with this broker
 and this port

i = 0
while i in range(4):
    status = ["operational", "docked", "maintenance"]
    data = {
        "speed":random.randint(300,1200),
        "battery":random.randint(0,100),
        "status":random.choice(status)
    }# creating random values for the data dictionary which is used as the message
    client.publish(topic, str(data))# publish the data as a string to the topic channel
    print(data)# unnecessary but to check the right data is being sent
    i = i+1
    time.sleep(2)# added this so that the subcriber could get the data and not be terminated{{
~
~
~
publisher.py [+]                                                    22,94         All
```

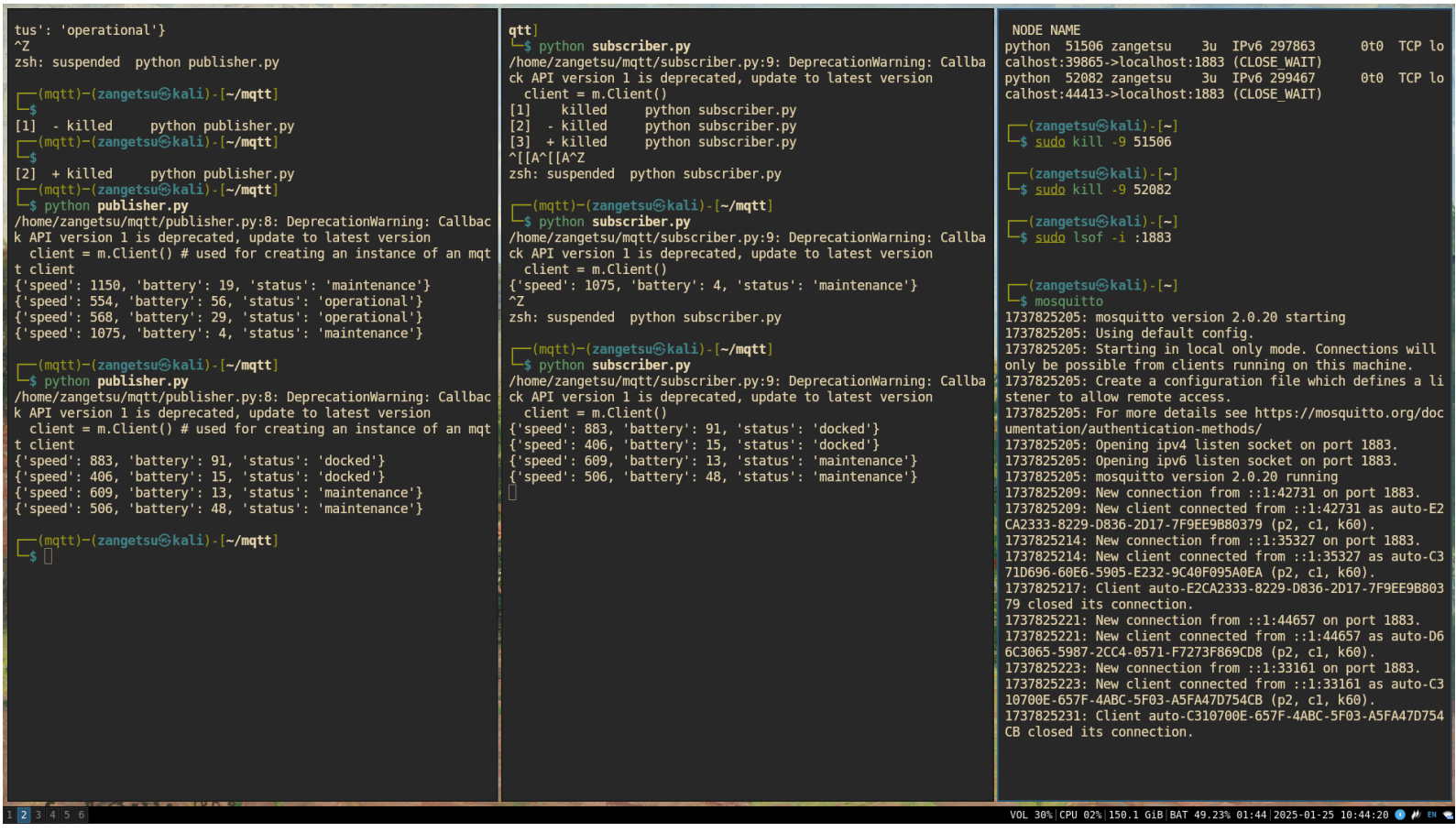
```
import paho.mqtt.client as m

broker = "localhost"
port = 1883
topic = "hyperloop/data"

def on_message(client, userdata, msg):
    print(msg.payload.decode())# a function used for convert the payload received by the subscr
iber into human readable string
client = m.Client()
client.on_message = on_message
client.connect(broker,port)# connect with that broker present on that particular port
client.subscribe(topic)# used to subscribe to that particular topic of the channel

client.loop_forever()#continuosly listen for the incoming messages from the subscribed topics
~
~
~
~
~
~
~
~
~
subscriber.py                                                       14,21         All
```

The above two codes are named as publisher.py and subscriber.py.
The publisher.py file is the one which acts as the publisher and sends the messages using random values.
Whereas the subscriber.py file is the one which receives the data sent by the publisher.
The notes are written as comments.
The output could be found by using the command mosquitto (which is the one that i'm using since it doesn't have any complicated setup)
Then in another terminal we must run the publisher.py file and the subscriber.py.
I runned the subscriber.py file first (obv) because this file must be first initialised to read the input from the publisher.py which would be runned later.
The output is



It is quite difficult to see(and I'm sorry for that ;\) but it could be seen that in the first terminal there are 4 random dict values formed(publisher) and the second terminal(subscriber) displays all the values.

If you’re wondering what the third terminal is, it is mosquitto running in the background and they are the output.