signalR allows u to create real time application on HTTP

HTTP is a standard protocol use to communicate between client and server

HTTP is stateless u make a request and u get a response server doesnt remem ber anything of the prevous request  
you have to figure out how to maintain these state eg cookies

signalR changes that and allows you to create real time applications,  
means something happens on server all client will know immediately

signalR creates a tunnel the connection remains persistent and there exists bidirectional communication between client and server  
between sever and client we have signalR tunnel

WHy and how SIGNALR  
4 technologies signalR uses to simulate real time communications  
1) websockets  
2) Event Source  
3) Forever Frame

4) Long polling

signalR is abstraction  
signalR is cross compactible - works in more browsers u can create a backend that will feed website as well as phone app and u only have one backend  
its not as easy as it is when used websockets

signalR has groups u can create a broadcasting system  
scability- SignalR works great even if there are lot of clients connected in open connections  
signalR used when you need full- duplex connection  
which needs low latency realtime connnection

**SNAT port exhaustion**1) NAT converts internal hosts private IP address to its public IP address got outgoing traffic  
2)also public IP address to an internal private adress for incoming traffic

SNAT maps the IP address of the backend to the public IP address of your load balancer  
SNAT prevents outside sources from having a direct address to the backend instances.  
every IP address has 65,535 ports  
Each port can either be used for inbound or outbound connections for TCP (Transmission Control Protocol) and UDP (User Datagram Protocol)

To establish an outbound connection, an ephemeral port is used to provide the destination with a port on which to communicate and maintain a distinct traffic flow. When these ephemeral ports are used for SNAT, they're called SNAT ports. By definition, every IP address has 65,535 ports

**Python server**

Display data visually for traders create the chart for the trader’s dashboard.  
Aim to generate a chart that displays the data feed in a clear and visually appealing manner for traders to monitor this trading strategy. Basically, you have to modify the existing live chart to be able to (1) track and display the ratio between the two stock prices (2) show the historical upper and lower bounds of the stocks' ratio (3) and finally, show 'alerts' whenever these bounds are crossed by the ratio.

Used a socket library at server side to establish connection 8080 port and simulated an excel file which acts as a datafeed so we have two stocks just for initial work with top bid and top ask price we calculated the ratio between the two and saw the if it hits either of the bounds if yes we generated an alert.

price = avg of top bid price and top ask price

ratio = ratio of two stocks say abc and def are two stocks

There was threaded to deal with multilple connections.

Daemon thread. Program exists even after threads are running

asset equal in client side we had testing module where we passed each quotes and assetequaled them .

**Azure cognitive search**

Azure is not search on azure cloud, search on your data. Bring your data you want to make it searchable you have a website you want to add this small text box

Search index is like a sql table, a *search index* is your searchable content, available to the search engine for indexing, full text search, and filtered queries

In Cognitive Search, indexes contain *search documents*. Conceptually, a document is a single unit of searchable data in your index. For example, a retailer might have a document for each product, a news organization might have a document for each article, a travel site might have a document for each hotel and destination, and so forth. Mapping these concepts to more familiar database equivalents: a *search index* equates to a *table*, and *documents* are roughly equivalent to *rows* in a table.

Entity and phrase extraction :

Key phrase extraction : evaluates unstructured text and for each record, returns a list of key phrases.

For example, in the text "*mumbai to delhi 32ft multi axle*"

Key phrase = [“Mumbai”,”Delhi”, *“32ft multi axle”]*

Entity recognition: the process of detecting the names entities as such as person names, location name, truck name

Entity recognition = {text:“ Mumbai”,category: Location, “confidencescore:0.97}

Fuzzy logic

We consider partial ratio to be >95

Used to deal with spelling mistakes , upper case lower case characters

Used fuxxywuzzy library and process module to find the match