## Question no. 3

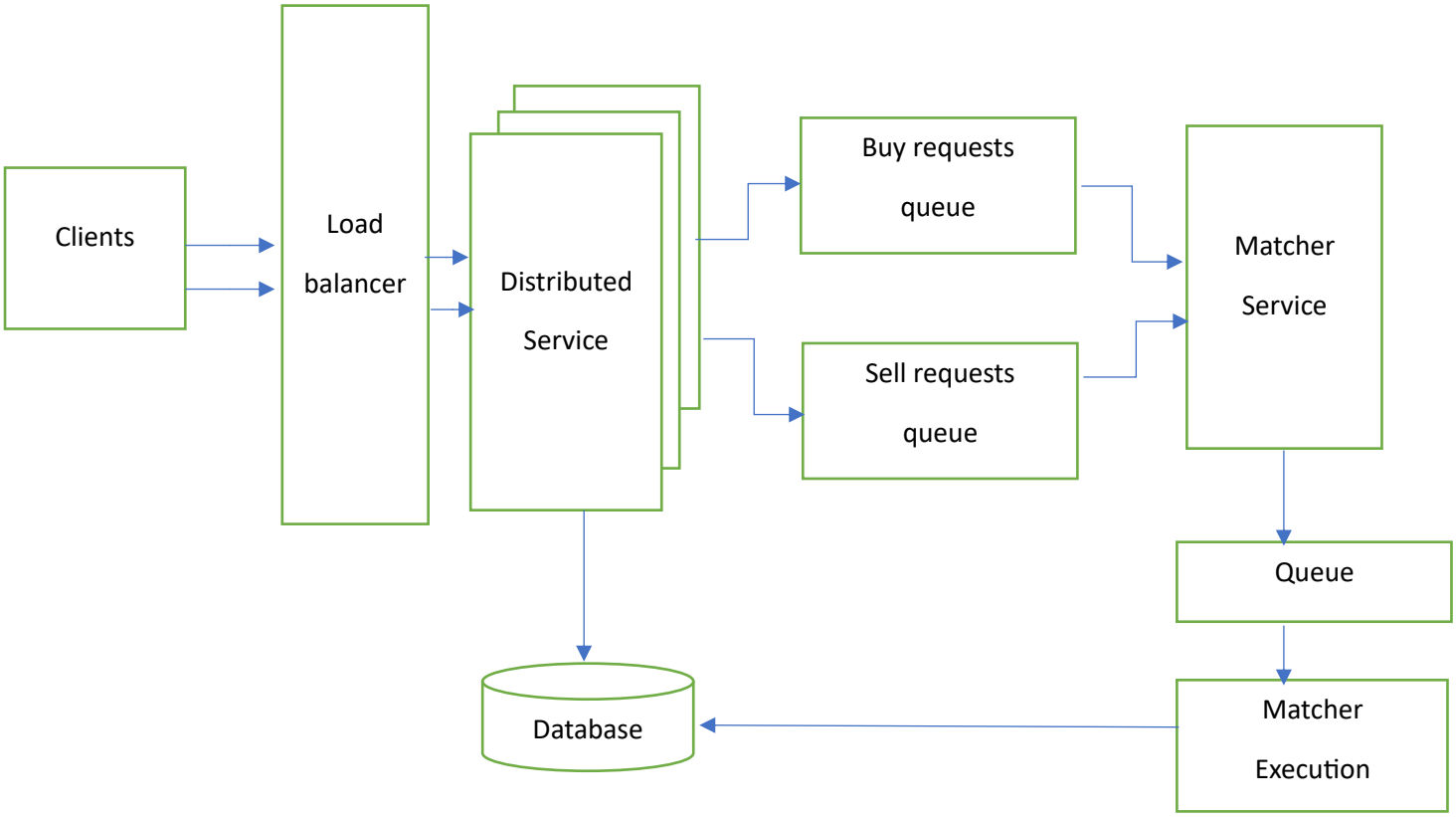
### KASB KTrade and Investify:

KASB KTrade and Invertfiy are applications that provide an online stock trading platform to the users – providing the users with an opportunity to start their investment journey:

- Stock trading is possible on the PSX using Investify and KASB KTrade. This may be a profitable strategy to invest in Pakistani businesses and benefit from the expansion of our country's economy.
- Commodity trading is also possible on the PSX using Investify and KASB KTrade. This may be a useful strategy for protecting against inflation and profiting from changes in the price of commodities like gold, oil, and wheat.
- Investify and KASB KTrade can be utilized to make future investments in Pakistan's stock market. This may be a practical approach to reach other financial objectives, such as retirement savings.
- Day trading (buying/selling stocks in the same day), Swing trading (buying/selling stocks over a period of days/weeks), and Position trading (buying/selling stocks over a period of months/years) is possible through KASB KTrade and Investify.

### System Design

A buy or sell order will be given from the client's side using a mobile phone, a laptop, a desktop computer etc. These requests will go to a load balancer which will balance the load of all the incoming requests among the distributed system – multiple hosts will be there that will take the requests balanced by the load balancer. These hosts can be employed in a virtual machine such as an EC2 instance. Thess requests will be stored in a database with their status. At this stage, their status will be in-progress. Then, these requests will be stored in two separate asynchronous queues – one for buy orders and one for sell orders. In each of these queues, we will have a queue for each company we deal stocks in to make the process a bit simpler. These queues should have a visibility timeout concept i.e once a buy request has been matched with the sell request, they should not appear in their respective queues. This matching is done by a matching service containing more than one matcher nodes – to increase the availability. If an order is not matched, they should be put back into their respective queues. Similarly, if an order is only matched partially, that partial matching should happen, and the remaining buy/sell stocks request should be put back into its respective queue. Then, these matchings should be put into a queue from where they are given to an executor service which will perform the execution of these matches. Once executed, the payment service will be called, the database of buy and sell request will be updated to change the status and the quantity that was fulfilled (partially fulfilled or completely fulfilled) for the order, and we can also use AWS SNS service to send notifications about the successful buy/sell order.




**API Architecture**

An online stock exchange can have internal and external APIs. Let's start with external APIs. If a user wants to sell stocks, an API is required for this process. This "sell" API might take as input the customer id, the id of the stock they want to sell, the quantity they want to sell and the price they want to sell that stock at. Similarly, another external Api might be a "buy" API that will be called once the user wants to buy some stocks on the online stock exchange website. This API would also take as input the customer id, the id of the stock they wish to buy, its quantity and price. Another important API might be the "get status" API that would be called when the user who either wants to buy and/or sell stocks, wants to get the status of their order.

For internal APIs, we can have for such platforms, a "matching" API that would match the buy and sell orders. Using this API, it would be possible to match a user who wishes to buy some stocks which another user plans to sell.

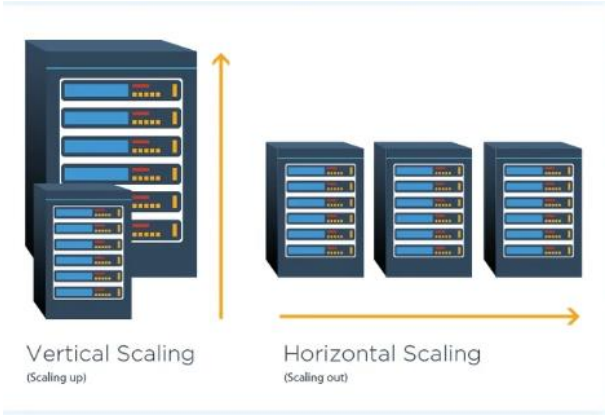
There is another "order management" API exposed by the API server that allows the system's backend functionality to be accessed by the web application. Log in and sign-up process can be done using an "account management" API gateway to use third-party platforms such as Google, Yahoo, Facebook to make users log in or sign up for the platform. For money transactions, we can have a "payment" API gateway that connects our system to banking systems to credit or debit a user's account based on whether that particular user sold or bought stocks.

For integration purposes, both Investify and KASB KTrade use Rest APIs. Rest APIs are used by Investify for integration with payment gateway etc. KASB KTrade uses Rest APIs to integrate with its system that is responsible for the post-trade processing of transactions etc.

**Aapproaches to scalability:**

The first and foremost approach to scalability is a load balancer to balance the incoming load among multiple servers. For KASB KTrade and Investify, the best approach would be employing an Application Load Balancer. An ALB would be beneficial for them as it is not only scalable, but it is also flexible and on top of that, supports a variety of features that are useful for online stock exchange platforms. These features include caching of content, user account authorization and authentication, and HTTP/2 support (a newer version of HTTP). Such a load balancer can be employed in the cloud using Amazon Web Services or Microsoft Azure. Doing this will increase reliability.

We can have vertical and/or horizontal scalability depending upon the incoming load as well. Horizontal scalability is basically adding or removing more servers to our system depending upon the incoming traffic. However, vertical scaling is modifying the power of the servers that are already a part of our system. These online stock exchange platforms can also use the concept of caching as an approach to their scalability. Frequently accessed data can be cached so that instead of querying the database again and again, the cached results could be sent back to the inquiring user.

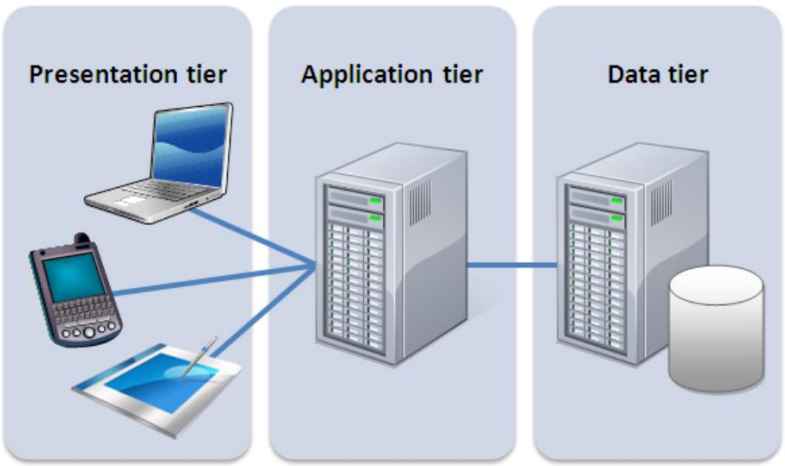



**Software architecture:**

Investify's and KASB KTrade's software architecture is based on a three-tier model – presentation tier, application tier, and lastly, database tier. This tiering makes it easy to modify one tier without causing troubles for other tiers.

For Investify, the presentation tier is responsible for handling all the user interactions. This tier has been designed using React and Redux. Processing user requests and communicating with the database tier are the responsibilities of the application tier. Python and Django are used to implement it. The PostgreSQL based database tier is responsible for the storage and retrieval of the data based on queries.

For KASB KTrade, the presentation tier uses Angular and TypeScript. For the application tier, the implementation of it is done by utilizing Java and Spring boot framework. The database tier has been implemented using Oracle Database, which makes the database of KASB KTrade highly secure.




## Strengths and Weaknesses of Software System Designs:

**Investify:**

Client-server architecture is the foundation of the software system design for Investify – making it highly scalable. The reliability of the three-tier approach for large-scale online applications is one of the biggest strengths of the software architecture used by Investify. Investify has several security mechanisms to safeguard user data. Moreover, the technologies used by Investify involve both open-source technologies and proprietary technologies, allowing flexibility and customizability.

However, the use of proprietary technologies can make the process of troubleshooting very complex. Moreover, as a three-tier modelling approach has been used in investify, testing it is more complex as testing such a model involves testing the interactions between different tiers.

**KASB KTrade:**

Like Investify, the biggest strength of KASB KTrade is that it is based on a three-tier model which makes the system modular by separating presentation, database, and application tiers from one another. Same as in Investify, it uses a combination of open-source and proprietary software programs, resulting in more customization and flexibility. Because its application tier is implemented using Spring Boot and Java, KASB KTrade is incredibly effective and has high performance.

The weaknesses of KASB KTrade's software system design involve all the weaknesses of Investify's software system design given above as both use three-tier modelling. Moreover, the security provided by KASB KTrade is not as strong as security features provided by other online stock trading applications. The order management system of KASB KTrade does not offer extreme reliability as orders, in some cases, are executed in an incorrect manner.

## Personal Experience:

**Investify:**

The interface for Investify is very user-friendly. The color palette chosen for the homepage is very soothing. The homepage also provides app screenshots which are very helpful for new users. Anyone wishing to invest in stocks can benefit greatly from looking at these screenshots. Given that the website offers a lot of information and guidance, it is especially beneficial for beginners. It provides a list of features, reviews, FAQs, contact information, and download links. I believe that Investify is a fantastic starting point for stock investment.

**KASB KTrade:**

The interface for KASB KTrade is also user-friendly as Investify. The homepage provides links to webinars and e-books for users so that the users can learn from experts and get guidance. Blogs are also provided to keep users up to date with the current trends. Their YouTube videos are also present on their homepage which are great for people who want to learn about the economy of Pakistan. Technical charting tools and real-time market data make KASB KTrade a very good platform for stock exchange.

**References:**


https://www.youtube.com/@system-design-by-vinayak

https://www.geeksforgeeks.org/what-is-a-distributed-system/

https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

https://www.webforefront.com/performance/scaling101.html