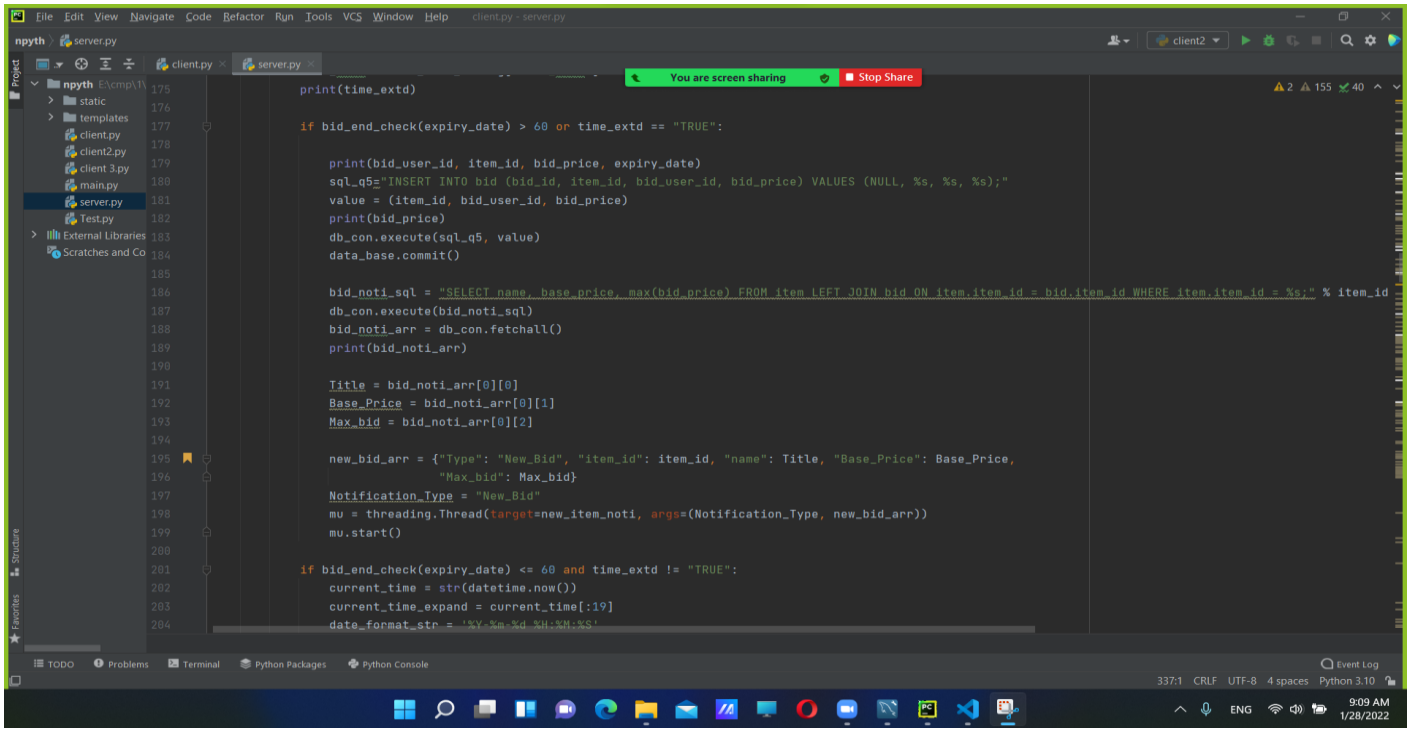
Mini Project

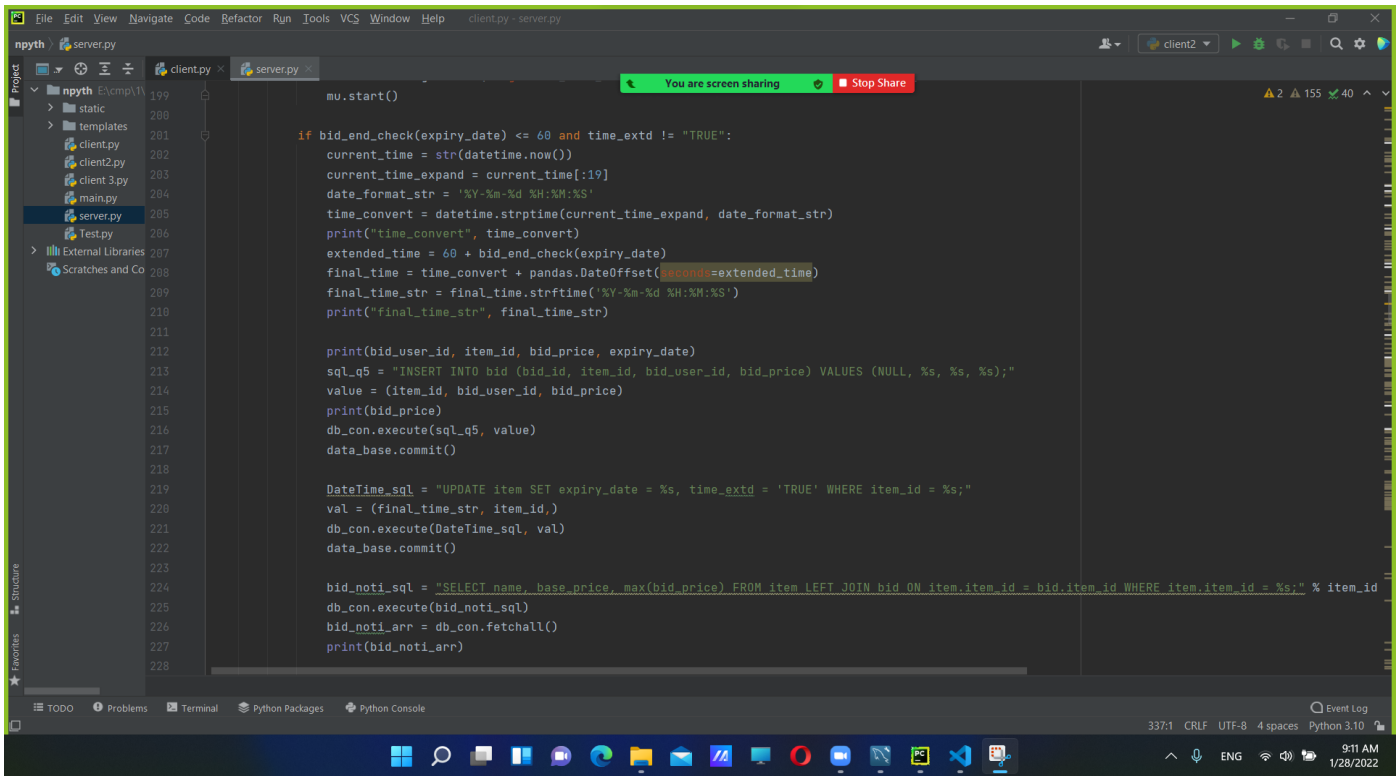
CCS112 - Communication Models and Protocols

Submitted by the Members of **Group 20**

	Name	**Registration Number**
1	Mohomad Iffath Hana(**Group Leader**)	21UG0092
2	Janith Kavinda Hettiarachchi	21UG0228
3	Kesara Mahima Lakpriya	21UG0166
4	Oshan Wathsala	21UG0771
5		

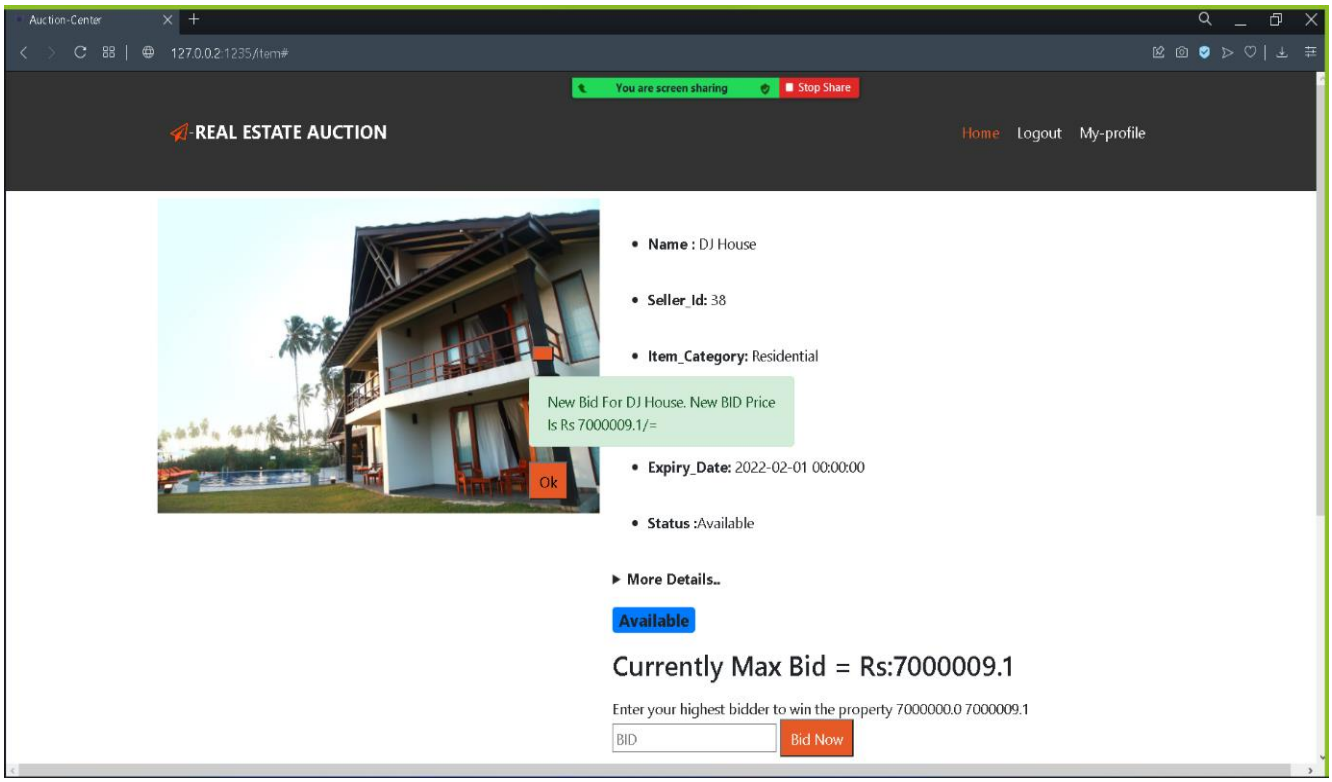
Design the protocol in a **diagram of the client-server model** highlighting the bid timing mechanism through the diagram or otherwise.





Design the protocol in a **diagram of the publisher-subscriber model** showing the information exchange between the publisher-subscriber and the auctioning server.

- The auction server sends continuous notifications to the subscriber till he or she confirms that the notification has been received. This is the way information interchange in the system we have developed. Screenshots of the above process are attached below.



Describe a mechanism to ensure that the client **ID can be kept unique** for each user.

- When a client logins to the system, an IP Address is generated for the client. This is used as the Client's ID throughout the session. Messages will be sent to the client based on the generated IP Address. This is performed by assigning each client with a new Python client file.

Clearly **justify why a publisher-subscriber model is suitable** for bid and profit updates (as opposed to client-server).

A publisher subscriber model is suitable for bid and profit update because publisher subscriber model, the broker notifies the subscriber about any changes on their subscriptions. In this method end points don't connect directly. Publishers connect with brokers. Subscribers subscribe to different things available as they interested in such as products and services. Brokers usually don't store data. They simply move data from publishers to subscribers.

In client server model nothing like this can be done. Neither the client can subscribe to the things they interested in nor he or she receives any updates about the things they interested in. Notifications won't be appeared unless the client logged in. Furthermore, client has to request in order to find out about updates regarding the products/ services. In every aspects client may find it annoying and difficult.

Therefore, publisher subscriber model is suitable for bid and profit updates.

Clearly state **functional and non-functional requirements** of the publisher-subscriber model.

Functional Requirement

Server notifies the subscribers about new biddings and new items.

Non - Functional Requirement

Subscribers are notified without any delay.

Notifications are popped up without any interference.

Explain how you use transport layer protocol to ensure these requirements.

UDP (User Datagram Protocol) is the transport layer protocol which used to accomplish above stated functional as well as non-functional requirements. The server sends a notification to the subscriber which is considered to be a functional requirement using UDP protocol

State the additional functionality you add in the application layer to fulfill the requirements not guaranteed in the transport layer.

The application layer is involved in communication handling. The server sends a notification to the subscriber using UDP. There is no guarantee that the notification has reached the subscriber. Therefore, we keep sending the notification until the subscriber confirms that he or she received the notification. This is the additional functionality we have added.