**IS 496 Computer Networks Project Report**

**Real-time Bidding System: A Reliable, Low-latency Communication Platform for Bidding Transactions**

Authors: Ken Wu (shwu2), Jack Chuang (yzc2), Thomas Huang (yenshuo2)

1. **Introduction**

Our project's objective is to create a real-time bidding system that will enable buyers and sellers to make network-based real-time bids on goods. All parties engaged in the bidding process will have access to a dependable, low-latency communication platform via the system. Buyers and sellers searching for a quick and safe way to engage in auctions are the application's target users. Our program will enable customers to bid on numerous items simultaneously in addition to allowing merchants to list their products and launch the bidding. The application will make the bidding process more efficient, provide up-to-date information, and maintain data integrity. This is crucial in high-stakes auctions since every second counts.

1. **Application Workflow and Implementation**

The application is designed with a user-friendly command line interface and efficient workflow to support smooth bidding operations. The main components of the application include user login, a bidding dashboard, real-time notifications, and secure data transfer.

*2.0 Getting Started*

To start the application, the server have to be launched first. This can be done by running a command in the terminal python3 server.py -hn *hostname* -p *portnumber.*

To start the client server, user will input python3 server.py -hn *hostname* -p *portnumber* -unusername in terminal.

*2.1 User Login*

Once users start the client server, they will be prompt to enter their password. The returning user will be ask to enter their password, while new user will be ask to enter a password and create new account.

*2.2 Bidding Interface*

The bidding interface have several option. Client will sends command to server and then server responds accordingly.

* BID: This command allows the user to place a bid on a specific auction item. Once enter the command, server will send the biddable items to client. The user would provide the item name and the amount they wish to bid, and the server would validate the bid and update the result accordingly. If the bid is successful, the server would respond with a success message. If the bid is invalid or unsuccessful, the server would respond with an error message.
* AUCT: This command allows the user to create a new auction item. The user would be asked to provide details about the item, such as item name and starting price. (Currently, the auction time by default is set to 180 seconds for testing purpose)
* GETALL: This command allows the user to retrieve information about all the auction items currently in the database. Each items has the information of
* EX: This command allows the user to exit or quit the bidding interface.

*2.3 Real-time Notifications*

*2.4 Socket Programming and Secure Data Transfer*

Our team developed the key networking functions of the application using Socket programming. TCP is used for secure and reliable data transfer during the bidding process. This ensures that all bids are accurately received and recorded, maintaining data integrity. It also enabled us to create a reliable, low-latency communication platform for buyers and sellers. In addition to TCP, we also implemented secure data transfer protocols to protect data transmission. We used encryption and decryption methods to encode and decode user data during transmission. This ensures that all data transferred over the network is protected. This further enhances the security of the bidding process. Overall, the combination of TCP and secure data transfer protocols provides a robust and secure networking infrastructure for our application.

1. **Technical Challenges**

There are several technical challenges we encountered when developing the application.

* Handling a large number of simultaneous connections
* Ensuring data integrity
* Minimizing response time

We resolved these challenges by carefully designing the networking functions, optimizing the application's performance, and using UDP and TCP protocols to balance speed and reliability.

1. If UDP will be used for real-time updates and notifications, how would you ensure that the users can get the updates/notifications? You may want to further consider the cost of missing a notification vs receiving the notification late and think about how to address them. 2. It's possible that the server may receive multiple bids at the same time for the same price. In such an event, you may need to establish a plan to resolve this issue. 3. The description of the external sources is somewhat vague. In your final report, you will need to have a more detailed description of the external resources along with the references.

specifically, you may wish to explore additional features and functions that could enhance the experience for both buyers and sellers. For instance, your app could prompt users to create an account that would enable them to easily track their buying, selling, and bidding activities. Moreover, users may wish to engage in multiple selling or bidding activities during the same period of time, and it would be interesting to consider such a scenario.

1. **Future Improvement**

There are few points we would like to improve if we have extra time on designing the application. First, adding front-end UI can significantly improve user experience with the app. User can bid items through graphical user interface, which will make their experience with the app more intuitive. Second, adding a database to store data will help improve data governance. Currently, we are only store bidding data, users data in txt files. With connecting the application with a database, retrieving data will be more efficient and reliable, as databases allow for faster search and retrieval of information, as well as easier data management and organization. Additionally, using a database can enable features such as data validation and concurrency control, ensuring the integrity and consistency of the data being stored.

* Customized auction time
* Time countdown

1. **Team Member Contributions**

* Jack Chuang: Networking functions using Socket programming
* Ken Wu: Application function and implementation
* Thomas Huang: Report write-up, demo preparation, and testing

1. **Conclusion**

Our real-time bidding system provides an innovative and efficient solution to the problem of real-time bidding. By using Socket programming and both UDP and TCP protocols, we ensure a reliable, low-latency communication platform for all parties involved in the bidding process. With our successful implementation and clear division of responsibilities, we are confident that our application can effectively serve its intended users and streamline bidding transactions.