

INHA UNIVERSITY TASHKENT

DEPARTMENT OF CSE & ICE

FALL SEMESTER 2018

SOC 3010 - OPERATING SYSTEM

TERM PROJECT REPORT

AUCTION

Submitted by

Anvarjon Yusupov	U1610026
Boburjon Iskandarov	U1610054
Doston Sukhrobov	U1610064
Oybek Amonov	U1610176
Rakhmatjon Khasanov	U1610183
Bokhodir Urinboev	U1610249

Group: 002

Junior



AUCTION

CONTENTS

SECTION 1: Abstract	4
SECTION 2: Introduction	5
SECTION 3: Project Overview	6
SECTION 4: Requirements definition	8
SECTION 5: Project Design & Implementation	15
SECTION 6: Results & Discussions	22
SECTION 7: Conclusion	29
SECTION 8: Future Work	29
SECTION 9: References	30
SECTION 10: Project Team	31

TITLE OF THE PROJECT:

Auction

PROBLEM STATEMENT:

It is becoming common thing that classic auctions turning into mess. That's why online auctions are being best solution to avoid chaos. So we decided to create small software application for auction. Our application can be really useful in our country as well, since in Uzbekistan people started to sell things at on-line auctions. The advantages of on-line auctions are as following. First, clients can participate in auctions from any place in the planet, even pay for the lots using their cards or making bank transactions without any problem, waiting or delay. Another thing is, it helps the organizers as well, no need to reserve a place, hire staff to serve clients. The only thing that has to be organized is to have a good server to cope with multiusers and good application that works flawless.

ABSTRACT:

Automation and changing the process to the digital world makes everything easier to handle. So, now let's see what happens beyond the cover of the system. Our system mainly handles the process of auction, like the work of the organizers are handled by server-side and clients are supported in client-side of the software. Server-side mainly handles the database manipulation, connection with clients, process input from clients, send/receive responds, and of course money transaction and process it. Client-side is much plain to understand in our software, as it handles only connection, process responds from server, send/receive responds, money transaction.

In other words, the admin (in our case, there is only server who covers for an admin) get request for the auction to the specific good, he/she adds the good to the database and server sets up the date for the auction and notifies the client about this. So the auction begins then starting price is announced, then users raise the price and wait till the end of the process. And so it goes on. We can claim that we can use our software in real world as nowadays even auctions and tenders are held on-line without going out the house. This is just demo version, further we can extend to the real alpha product adding additional features and improve the stability and performance.

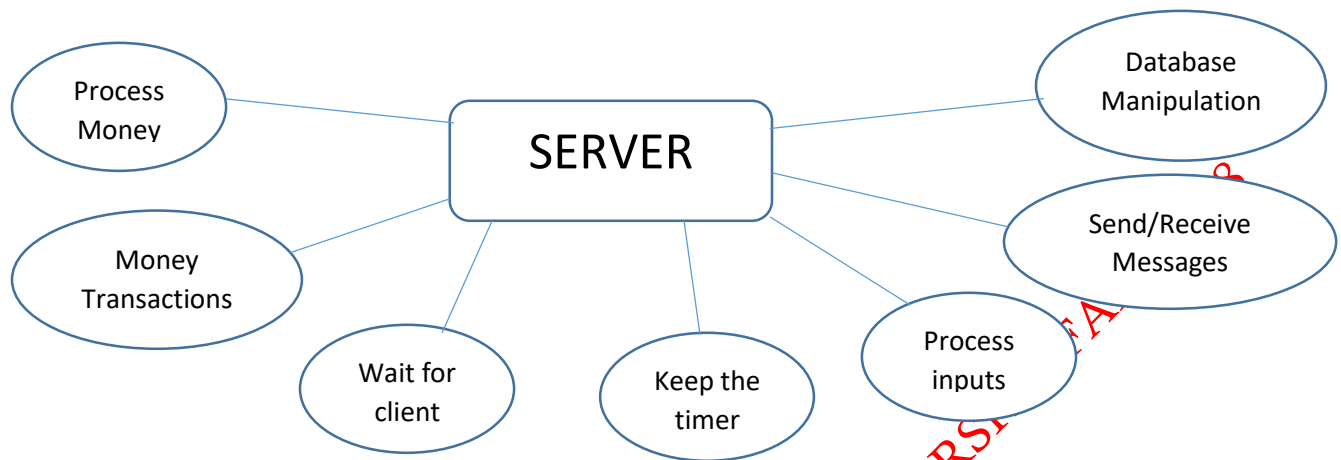
INTRODUCTION

From the period of 1995 to 2000 online auctions gone from nothing to the functional methods used by the millions of people from various industries. The factual example of the development of this system was introduced by the information list provided by Yahoo in 1998 which explained that the number of the automatized auctions were about the 90 and in comparison to the number in current time the list is much bigger than existed ones. Auctions made great success among the other forms of electronic commerce methods which is used in the industry. The functions that does exist in online auctions has obvious advantages over the local off-line ones as their functional process does not follow with automatized steps and highly vulnerable for the simple human factors and auction process physical destructions.

Auctions support dynamic formation of the prices, thereby enabling exchanges in situations where a fixed price unless it happened to be right which does not support many deals. The online environment is conducive for the auctions in an online environment because of the following properties. The network supports dynamic communication which makes the management of the protocols involving unknown number of participants. Unlike physical concentration of misunderstandings and shout at the room made by the traders the network auction mediator can have direct access on the information of the participant and control with the process within the auction rules. Furthermore, the automation process of the negotiations and bit control process which can be represented with a help of computational process. The case with online auctions make the process to be running in a real-time as in sense that off-line auctions tend to be which makes the identification of the participant easier and faster broadcast of the bids message (multicast communication protocols) Therefore, the widespread of the online auction protocols are not surprising in current moment.

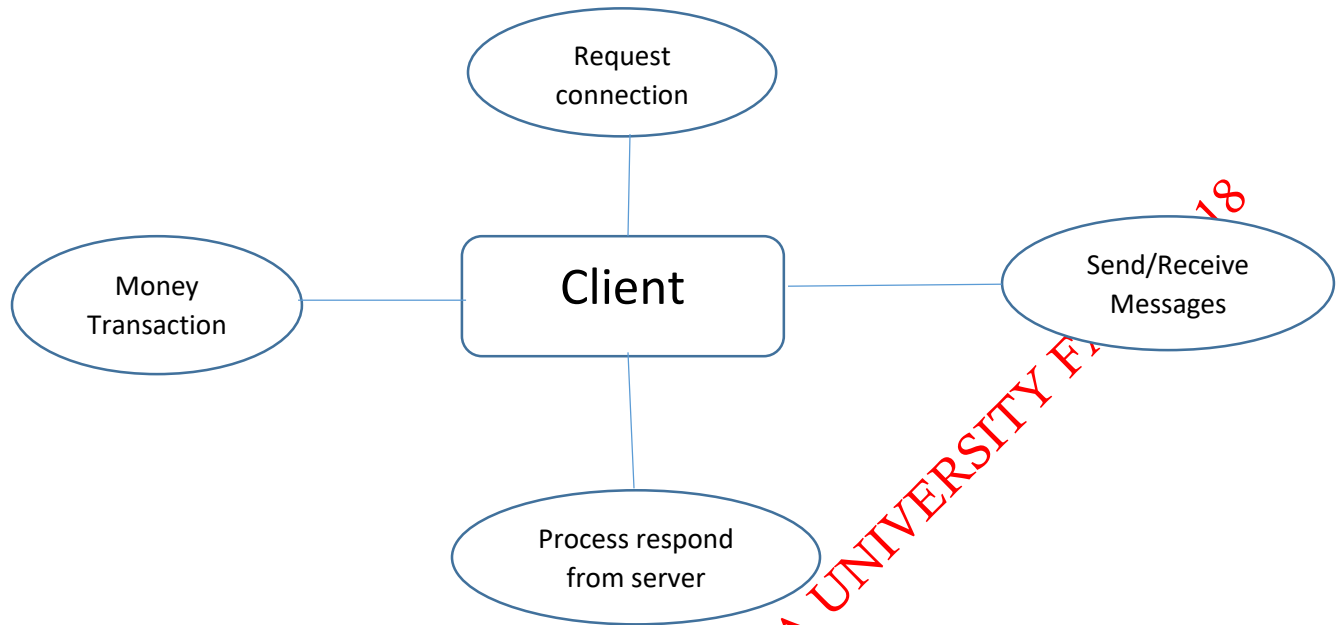
All the functions including the participants' identification in the beginning of the auction and the message notification while bid accepted or rejected by the participants and real – time monitoring control over the process of the auction procedures included in the system of the application that our team presenting in this report.

OVERALL PROJECT OVERVIEW:



Server-side

1. Database Manipulation - all the information on the goods are stored in server database. When the session starts all information is accessed and be ready to process. We used SQL server to hold the database. The database has the following tables: visa_card(for money transactions operation journaling), bid(for recording information on successful biddings and the bidders), lot(information on lots are stored in this table), and client(information on clients are stored here).
2. Wait for client – as soon as server is in running state, it will wait for client connection. As in auction new client can come at any time, wait for client function will always wait for new clients.
3. Send/Receive Messages – our application will be working on messaging concept, all the time server will receive responds from users, and send new requests for users on new prize(bids), acknowledgement about auction process and goods.
4. Process inputs – as soon as server receives responds from user, it processes the input, validate it (if new prize is more than previous), if no new prize is not proposed this prize is accepted.
5. Keep the timer – when prize is proposed by client, time is now checked. If no increase in prize is received by server, the good will be sold.
6. Money transactions – as soon as the good is sold, money transaction process starts, server will wait for the right client to send money.
7. Process money - when received, the money will be processed in server side, taking some interest rate, and send to good owner.

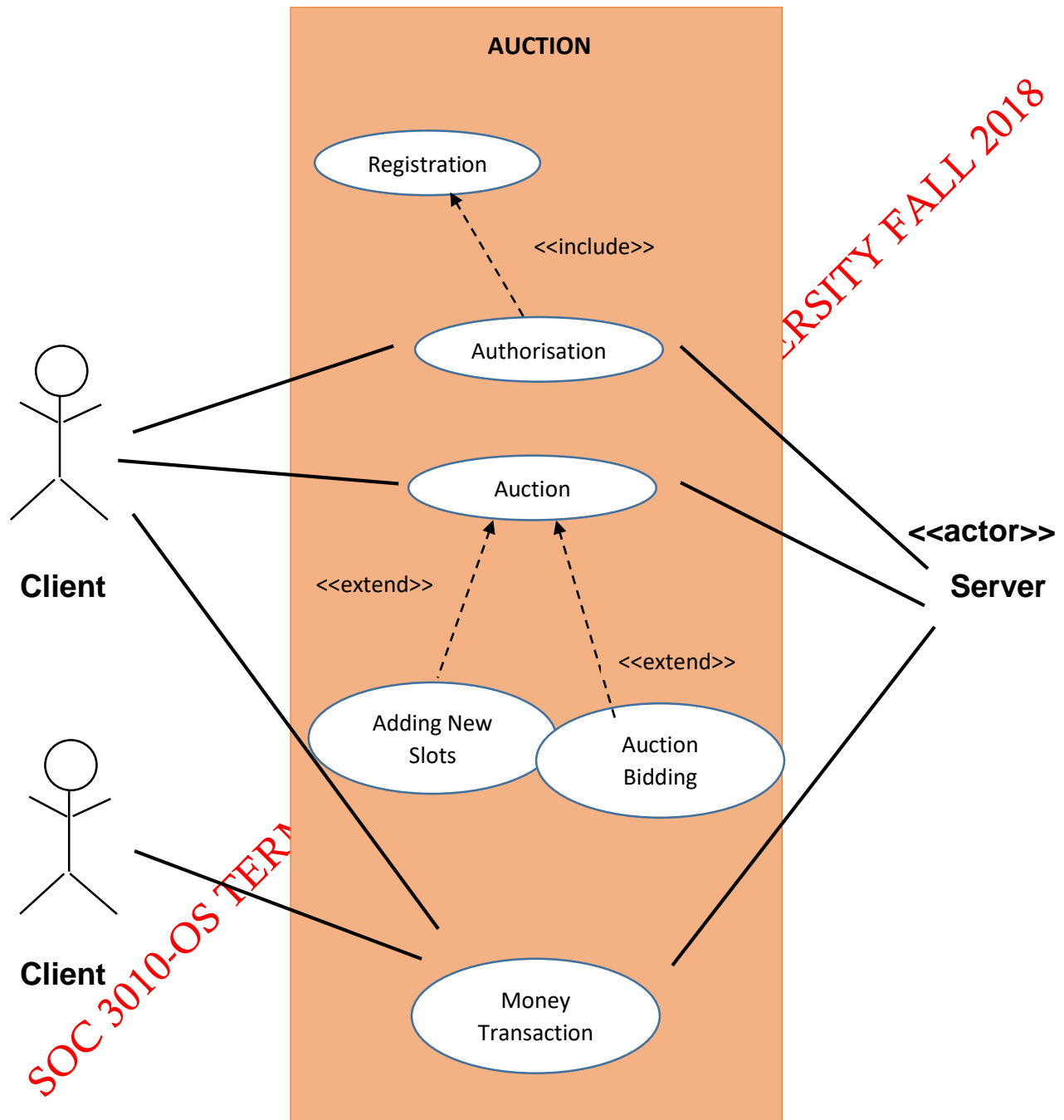


Client-side

1. Request connection – to participate in auction user has to get connected to the server first, to do that request connection to server.
2. Send/Receive Messages – client listens to the server to receive about auction acknowledgement, prizes, goods information and etc.
3. Process responds from server – when information is received, it should be displayed to the user, and process the respond back to the server as soon as client decides to accept/refuse to buy the good
4. Money Transaction – if client buys the good, money should be sent to the server-side so the money is delivered to the good owner.

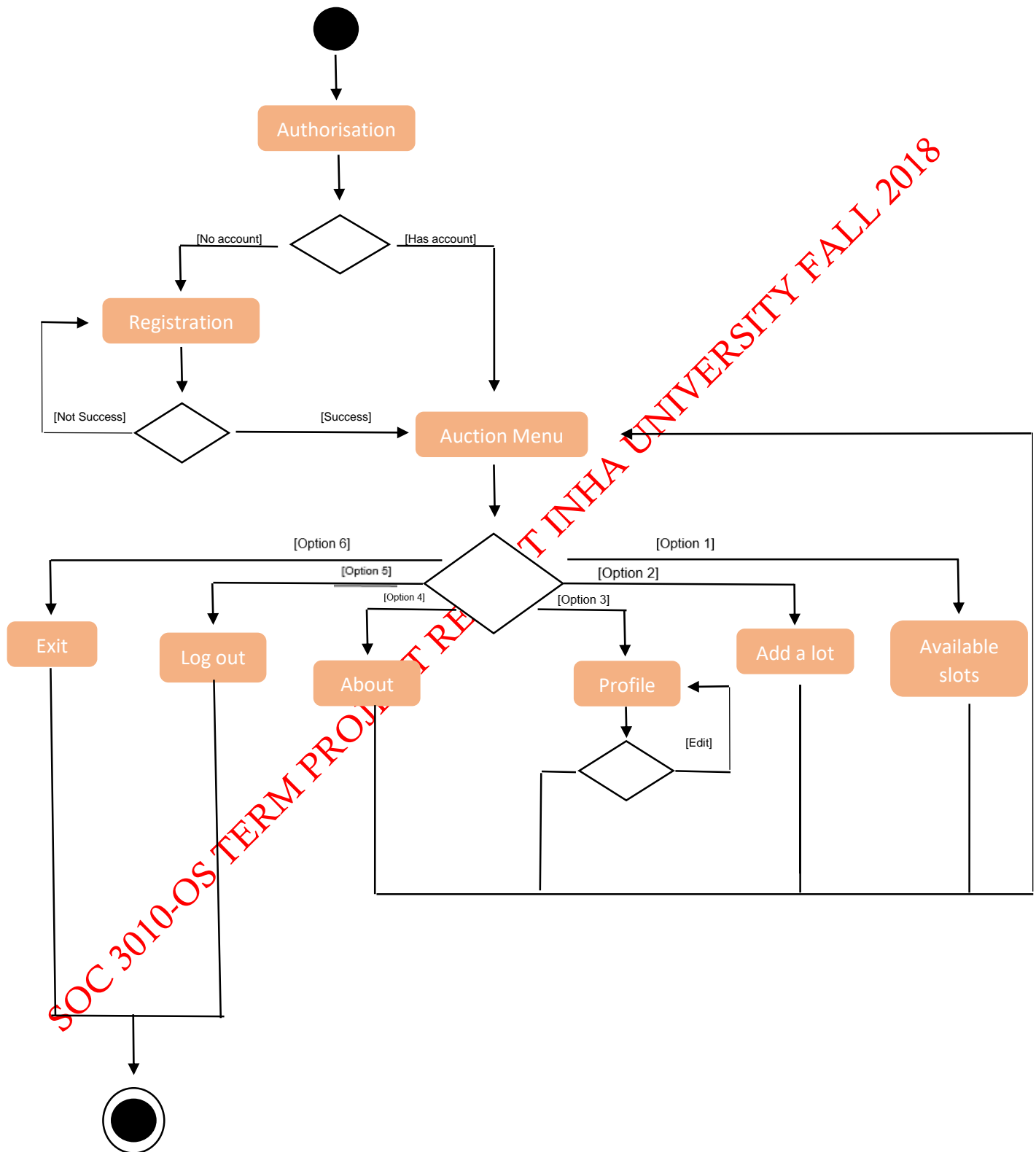
REQUIREMENT DEFINITION:

USE-CASE DIAGRAM

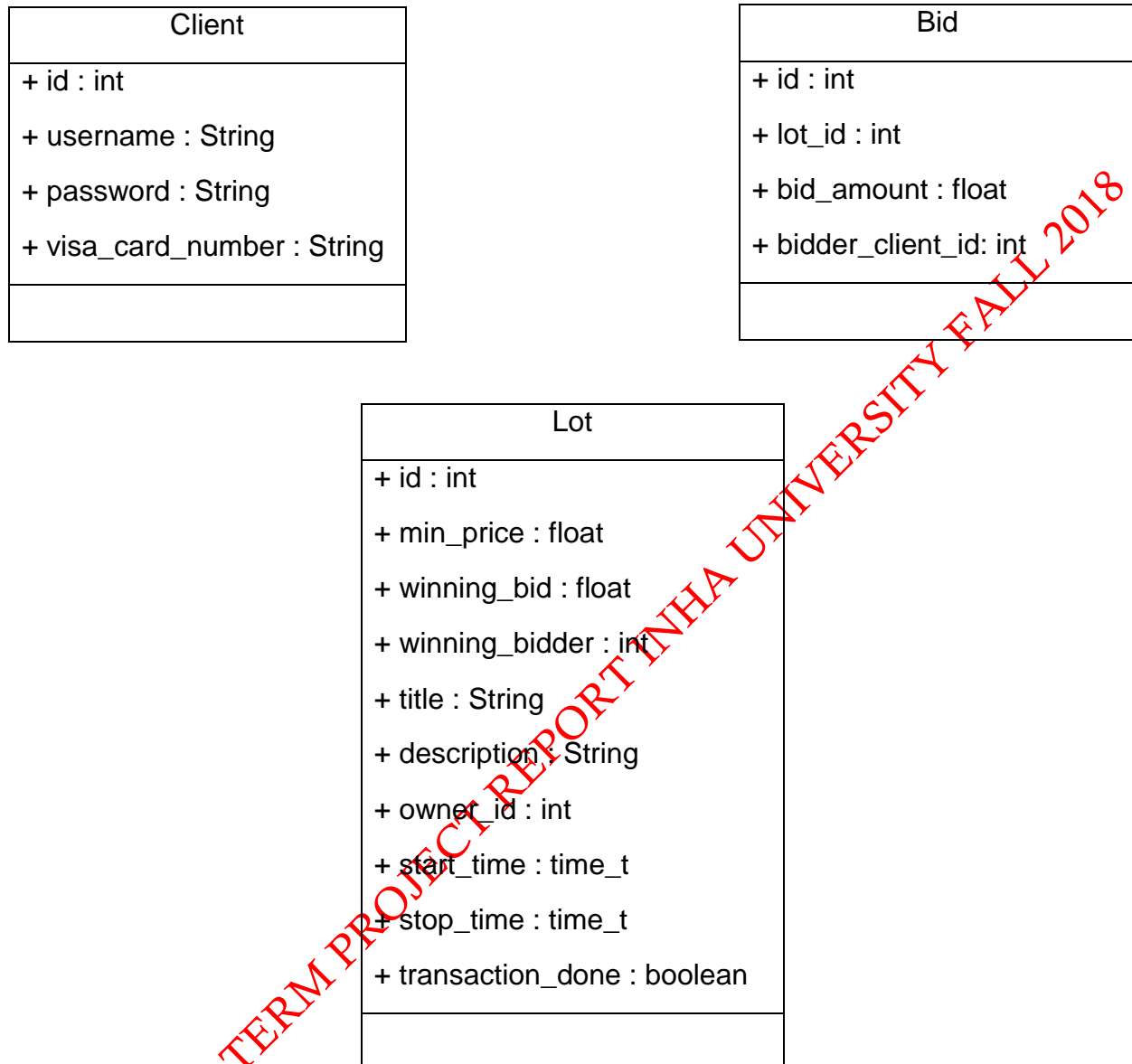


1. The use-case diagram above describes main features and operations of our system called *AUCTION*. So, there are only 5 main functionalities that has to be carried out. They are *Authorisation*(*Registration* included and *Login*), *Money Transaction*, and *Auction*(*Adding new lots* and *Auction Bidding*) itself. *Authorisation* has two actors namely *CLIENT* and *SERVER*. The *Authorisation* use-case describes the process of the logging into the application. However, we all know that in all application one can register themselves as customer. Thus, we need a registration as well, so *Authorisation* includes into it the *Registration* too. The *Auction* use-case has the same, *CLIENT* and *SERVER*, actors. *Auction* use-case includes *ADDING NEW LOTS* and *AUCTION BIDDING* into it. The last use-case *MONEY TRANSACTION* has three actors, two clients *OWNER*, *AUCTION PARTICIPANT*, and *SERVER*.
2. The activity diagram below describes the whole process where a client is offered the services we provided. Firstly, client has to pass the authorisation part. Assuming client has an account, he/she can successfully pass the authorisation part. A client can choose to register unless they do not have an account. A registration process will last until a client can register successfully. Registration may fail if the username of a new client is already existing in the database. After a successful authorisation, client is taken to the main menu, where user has 6 options. User can choose either to ask for available lots, adding a lot, see a profile, or a *ABOUT* part, and to *LOG OUT* or *EXIT*. Choosing first option user can see all lot for sale and can participate if wish. And, later come back to the main menu again. Option 2 offers the users to add their lots for sale, they are asked to provide some information on that and confirm the request. User is then taken to main menu again as in previous option. Option 3 enables the capability of showing the user their profile information which is requested from the server if chosen this option. User can either go back to main menu or choose to edit their profile information. Option 4 is given to see information on the organization and developer's team. Last two options, namely 5 and 6, are given to come out the application. Choosing *LOG OUT*, user is logged out and taken to *AUTHORISATION* window. When a user chooses the *EXIT* option, application is to be terminated.

ACTIVITY DIAGRAM



CLASS DIAGRAM

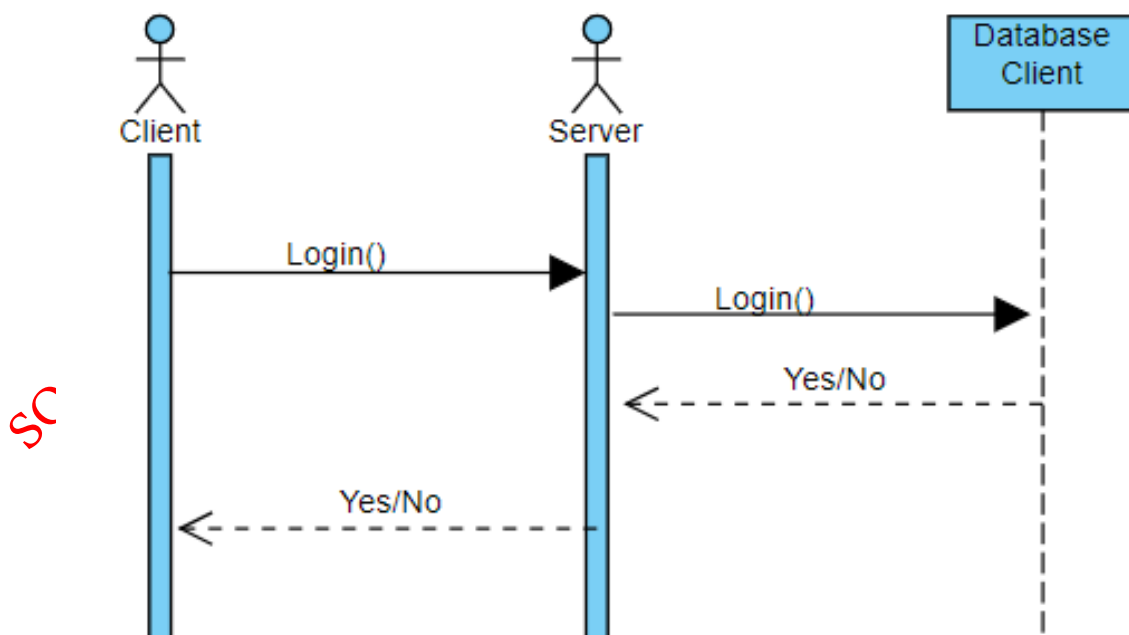
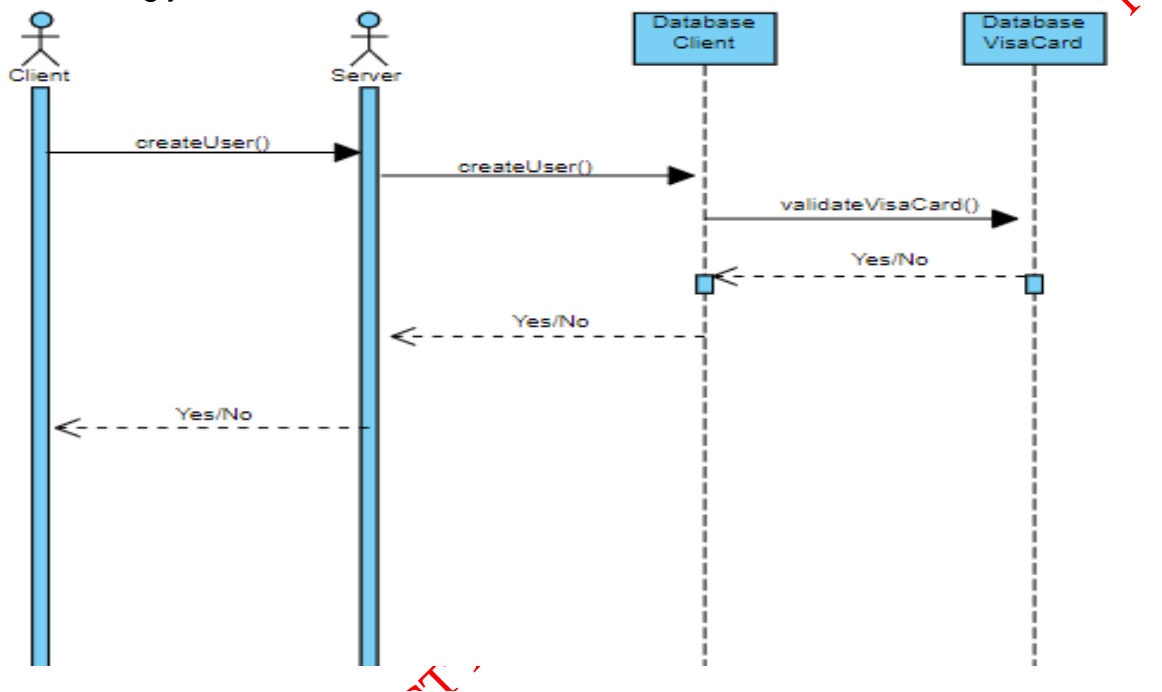


3. The above class diagrams show the structure we used to define our real-world entities. In our auction software, we work with clients, their lots, and the information on bids. So we have created above defined structures to store them. Client has own id, username, password, and visa card to pay for the lot they bought. When it comes to bidding, we need to know the which lot is being sold, its price, and who offers the price. The biggest entity in our diagram is lot itself, as you can see above. When a lot considered, it has its own id, minimum price, price of selling, who offered the price, information on lot (title, description, time for auction), and whether money transaction has been made after the auction.

SEQUENTIAL DIAGRAM

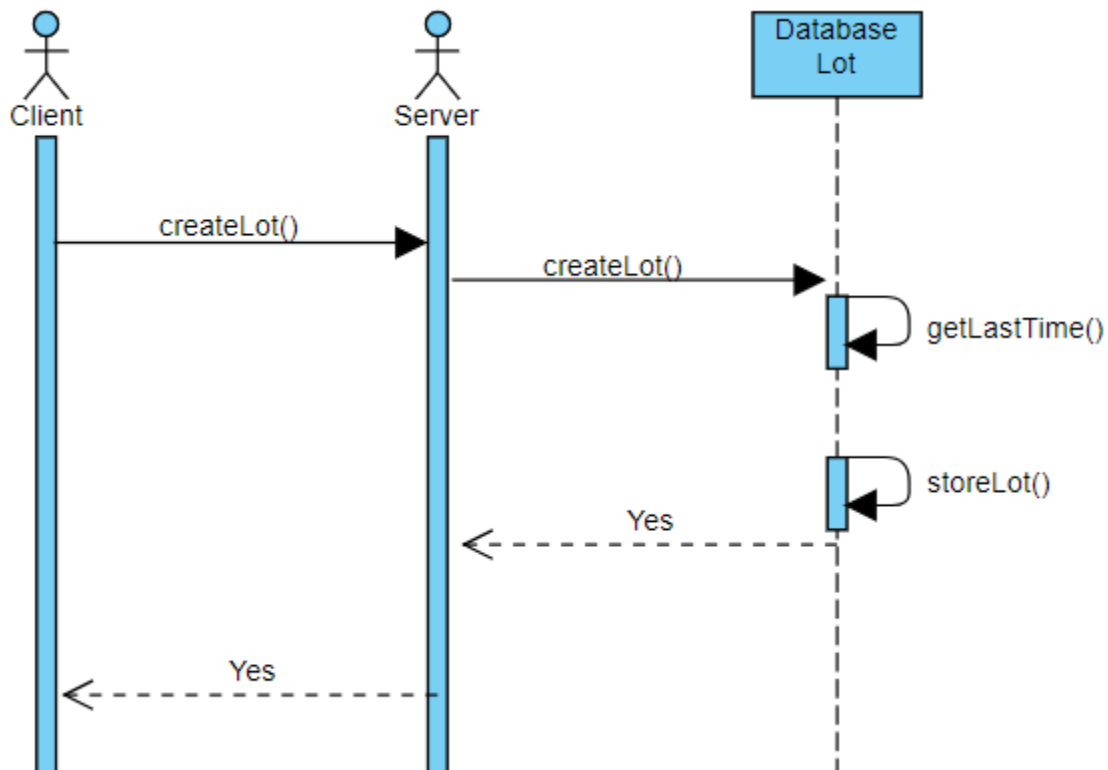
4.

- 4.1 The below diagram depicts the behaviour of the client and server when a new user is to be created. Client requests the server to accept it as a new user. Server in turn checks the database for ambiguity between username, and whether the card of the user is valid. Then, Server responds to the client accordingly.



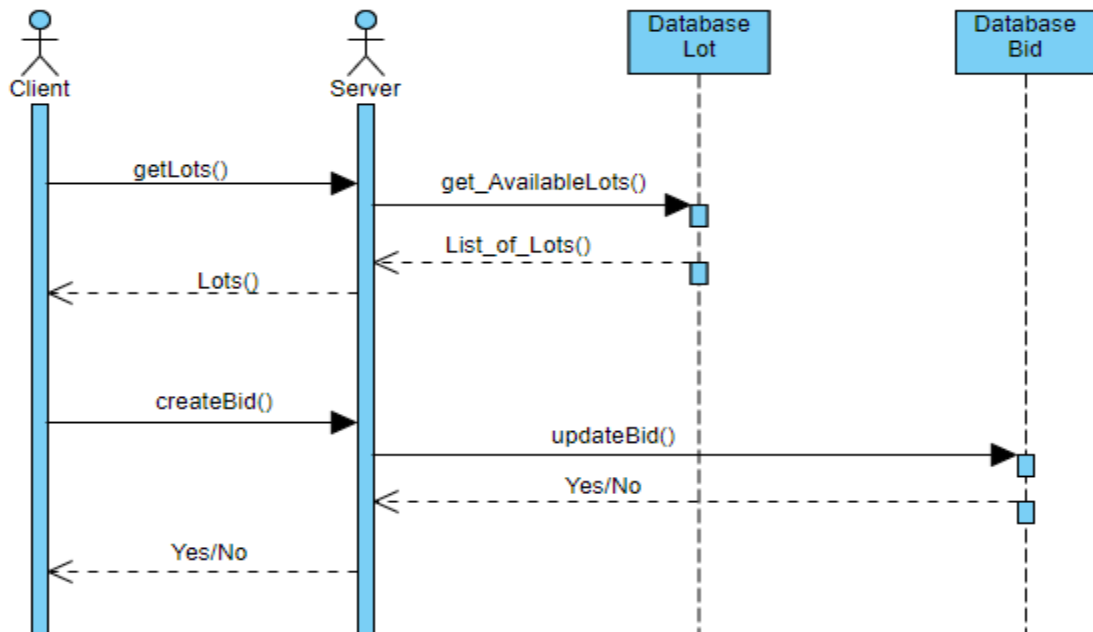
4.2 Above diagram is for logging in process. User inserts the login information, and requests for the login. Server checks for the existence of the user, and responds to client accordingly.

4.3 Sequential diagram below describes the behavior while creating a new lot. Client requests for accepting the lot, and server takes the lot information and assigns the lot time according to the earliest possible time after checking the last auction time.



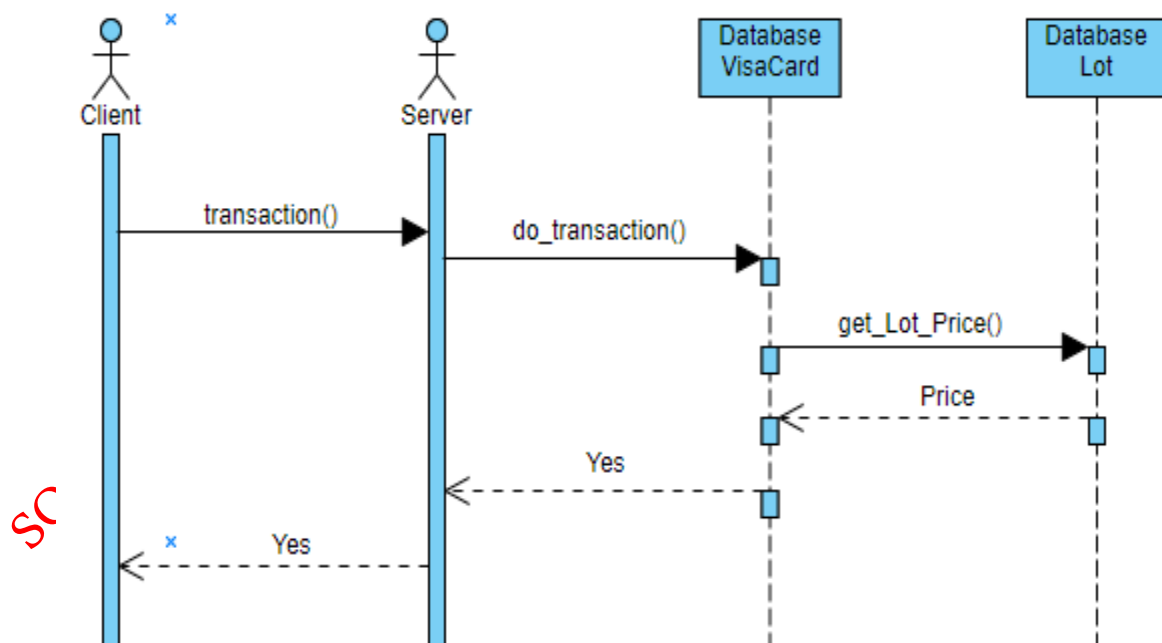
4.4 The behavioural diagram below shows the sequence of operations performed while actual auction happens. User asks for the available lots to choose. After choosing the lot, user takes part in an auction and possibly bids. Server handles the bidding after getting the request from the client

SOC 3010-OS TEST PREP



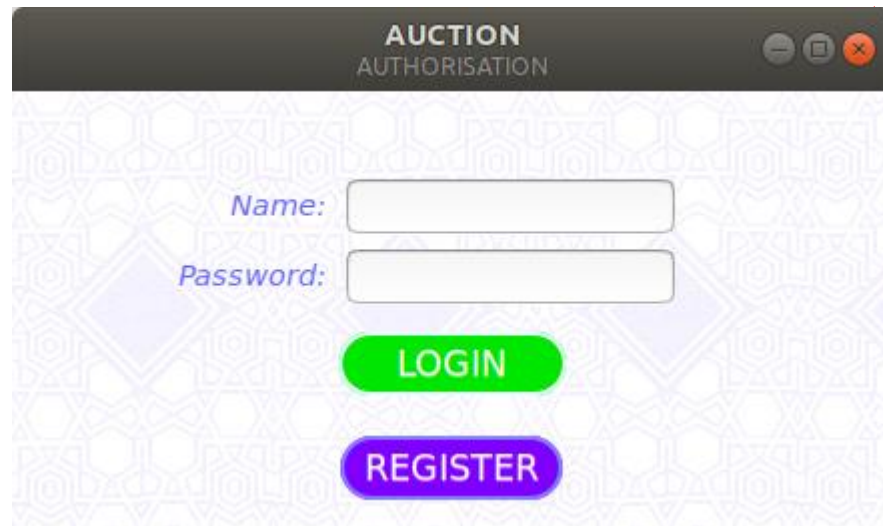
18

4.5 Below diagrams describes the process of money transaction between a server and a client after an auction. User requests server for the transaction to occur. And, server does the job by accessing the database and calculating the money to be transferred to the lot owner and the amount to be charged



PROJECT DESIGN AND IMPLEMENTATION:

Design of the project was made with a help of GLADE and GTK (graphical toolkit). The graphical user interface of the application of the client side was fully written in C language and was linked with CSS (stylistic user interface scripting language) standard. The utilization of the GTK (graphical toolkit) was realized thanks to the libraries that does exist in documentation of the toolkit. The functionality of the client side gives an opportunity for the users to control over the processes of the lot availability and their personal information. The first side that users face while using the application is the identification window.



This step of identification is divided in two parts from which the first is “Login” part which is used for checking the registration of the existing user information. The “Authorization” window gives two input boxes for the user one is “Name” and the second is “Password” where the user enters the information which is required for the authorization of the existing user (only in case of the user’s choice in “Login”). When user enters particular information in the fields given by the application checks the database for the existence of the authorization of the user, after the checking database there will be two choices from which the first is existence which makes the application to open a new window of the “Menu”, in case it does not identifies the user it will give page fault, in this case the user will be asked to enter the information again.

In case of the choice with “Register” button the user will be asked to enter the information for adding it to the database.

AUCTION
REGISTRATION

Name:

Password:

Confirm Password:

Card ID:

REGISTER

In case "Login" is identified by already registered user, the new window of the "Menu" window will be opened.

AUCTION
MENU

AVAILABLE LOTS

ADDING A LOT

PROFILE

ABOUT

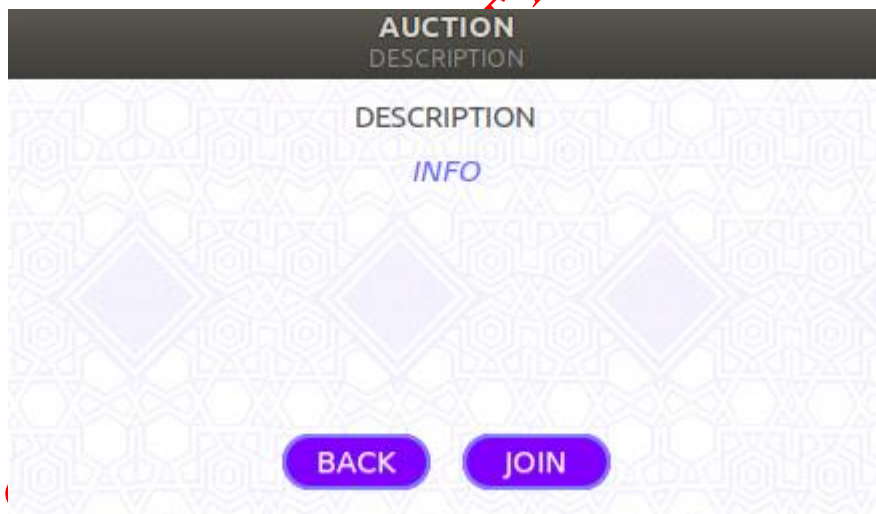
EXIT

LOG OUT

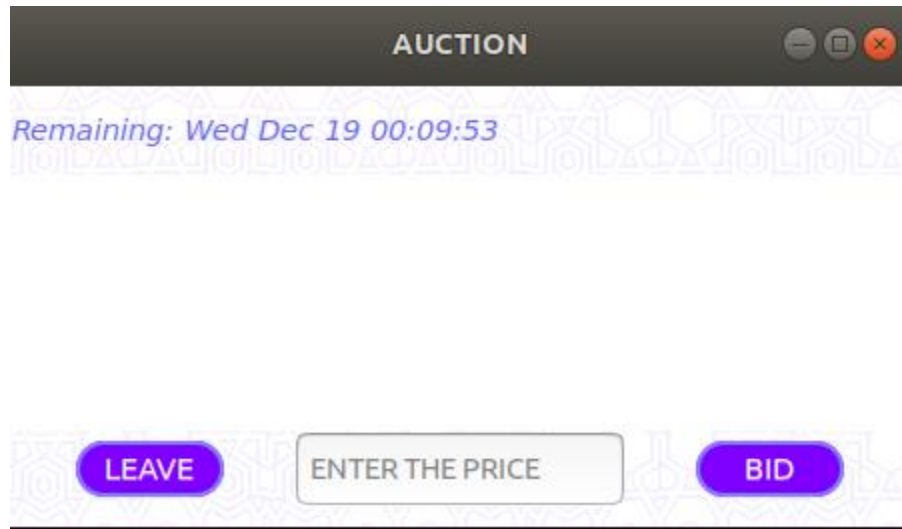
The menu window consist of "Available lots","Adding a lot", "Profile","About", "Exit" and "Log out". "Avaiaible lots" give the user an opportunity for controlling the process over the availability checking within the process of auction. And, assuming the client chooses this option, all lots for auction are received from the server and displayed on the screen as a list. Client chooses the lot him/herself, and auction is held in auction view for 5 minutes. Provided no bidding occurs in 5 minutes, lot is then considered not sold and probably, auction will be organized later on client's wish.



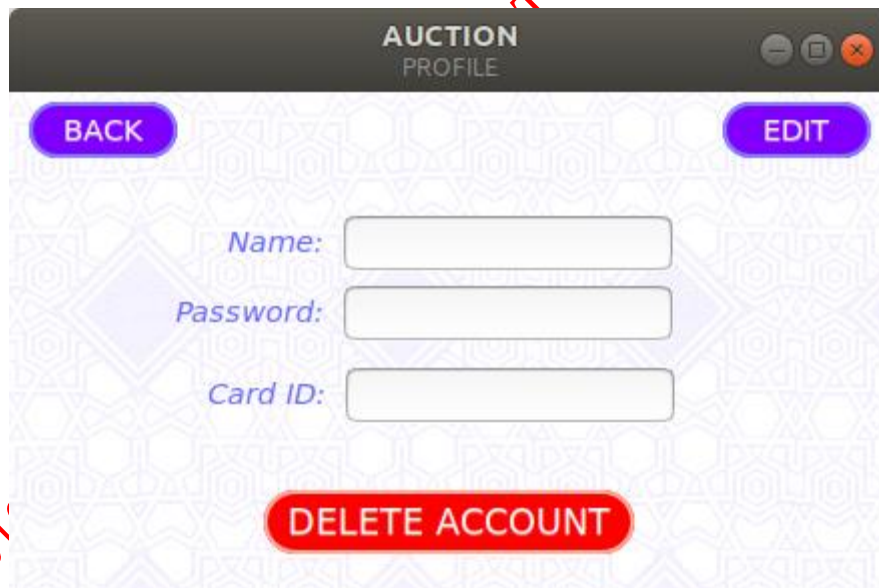
In above window, we see so called "Available Lots". It is used in a such way so that customer has all opportunities to choose the lot in list of various lots. Right now it contains 4 lots. With their tittle, starting time and minimum price that was given by customer (starting price of lot). By clicking to MORE button, we are allowed to have access to description of particular lot.



In the above window, information on the lot will be given, and user has a choice to participate in an auction for the lot or go back to the available lots list. In case, the user decides to participate in the auction, the auction window will be shown (refer to figure below). At the top, user sees the remaining time, in the middle there is area for information on bids of other clients. And, below of the window, there is area for offering client's bid. *BID* button is clicked to offer a price, or *LEAVE* button is clicked to leave the auction.



In the "Profile" section of the "Menu" user is able to check the information which was provided by the user himself/herself beforehand. In this window the user is able to modify, add or delete the information of the user.



The image displays two screenshots of a mobile application interface for an auction system.

The top screenshot shows the "AUCTION PROFILE" window. It features a dark header with the title "AUCTION" and subtitle "PROFILE". Below the header are two purple buttons: "BACK" and "OK". The main area contains four input fields: "Name:", "Password:", "Confirm Password:" (with a masked password "*****"), and "Card ID:". At the bottom is a prominent red button labeled "DELETE ACCOUNT".

The bottom screenshot shows the "AUCTION ADDING A LOT" window. It has a dark header with the title "AUCTION" and subtitle "ADDING A LOT". Below the header is a purple "BACK" button. The main area contains three input fields: "Title:", "Min Price:", and "Description:". At the bottom is a green button labeled "ADD".

The window above is the GUI for an adding lot option of the main menu. User can input the required fields and press *ADD* button. And, user is taken to *CONFIRMATION* window, where a user is asked to confirm the operation after getting familiar with the *TERMS & CONDITIONS*.

AUCTION
 CONFIRMATION

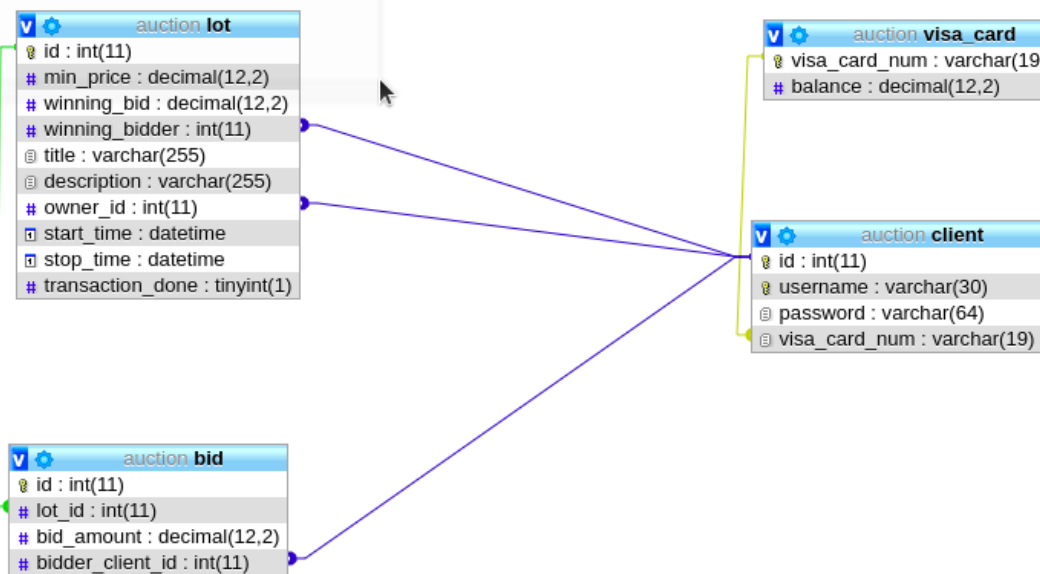
TERMS AND CONDITIONS

1. Any lot that is added can not be cancelled
2. Any lot that is added can not be edited
3. We can not guarantee successful trade for the lot
4. We guarantee that added is never deleted unless it is sold
5. We guarantee that the lot will be sold at least for minimum price that client mentioned
6. Each lot that is sold charged by 3% of the lot price

☐ Yes, I fully agree

Confirm
Decline

In current window above, we see terms and conditions that are followed when lot is added by customer. Confirm button is sensitive only and only if whenever toggle button ("Yes, I fully agree") is toggled. And after clicking confirm button client side sends to server all information about lot. Which are *Title*, *Minimum Price* and *Description*.

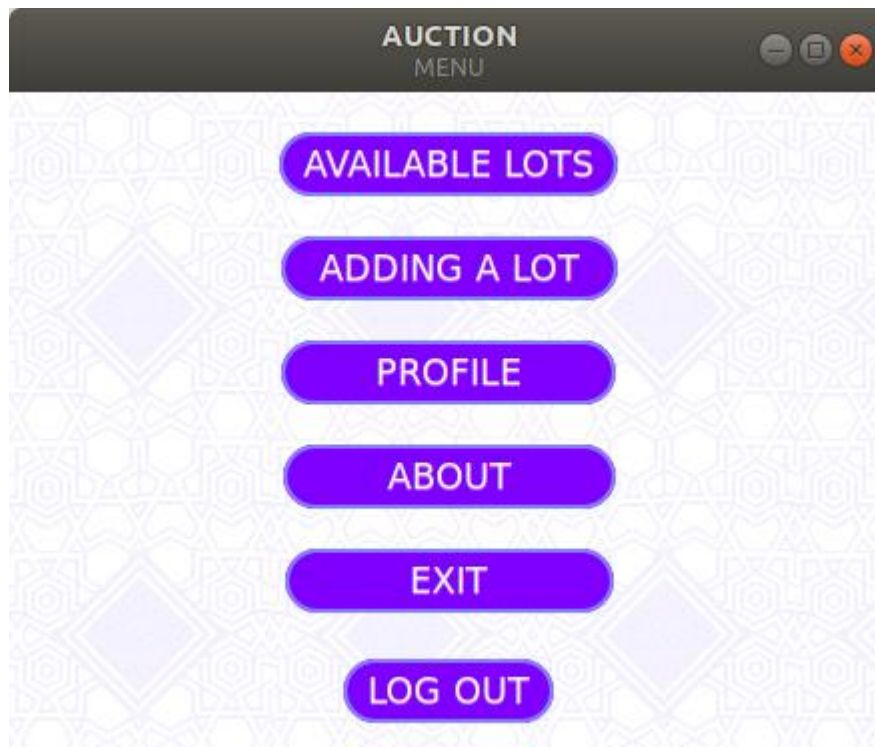


Design of the database and its implementation has been achieved using MySQL technology for Linux operating system. The overall picture of the database tables and relationships between the tables is shown in the below picture. As you can see, we have four tables, since we do not need to have a big and complicated database. We have separate tables for visa cards, bid, lot, and client. The following relationships are drawn to keep the relational integrity between tables. The relationship between *visa_card* and *client* tables is achieved by fields *visa_card_num* and *visa_card_num* respectively. And, using client's *id* field, bid and lot tables are reference (*bidder_client_id* in bid and *winning_bidder*, *owner_id* in lot table). The database system we created handles all possible cases, like UPDATE, INSERT and SET operations in SQL.

Inter-Process Communication (IPC) which is namely socket programming part. Socket of the application we are developing is as similar as in any other socket servers, it does work in a way that system call *socket()* created, after *setsockopt()* system call, after Bind the socket to an address with a help of *bind()* within which client can connect with server and server will be ready to listen and accept the information from the client and their work is done in *listen()* and *accept()* respectively. The application does use the TCP (Transmission Control Protocol), which accepted as the one type of stream socket that is accepted as a reliable connection oriented protocol. In this protocols the application creates the channels across a network. In addition to it, the channel manages how the message into small packets before their transmission over the internet and reassembled in the right order at the destination address. At the same time each time gateway forward the determination of the IP address. When we created the socked we used UNIX domain as an address domain with making the domain to be the same type and to have the same domain. With this communication process common file system which we used for share between each other at the same time the communication domain can be addressed also with internet domain as a usage of Transmission Control Protocol makes it possible for the other devices to collaborate within one other.

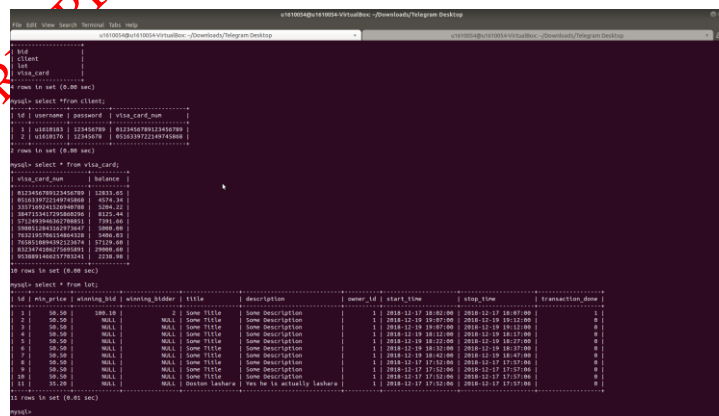
RESULTS & DISCUSSION:

The fully functional system at the end has an entrance look like in the picture as follows



FALL 2018

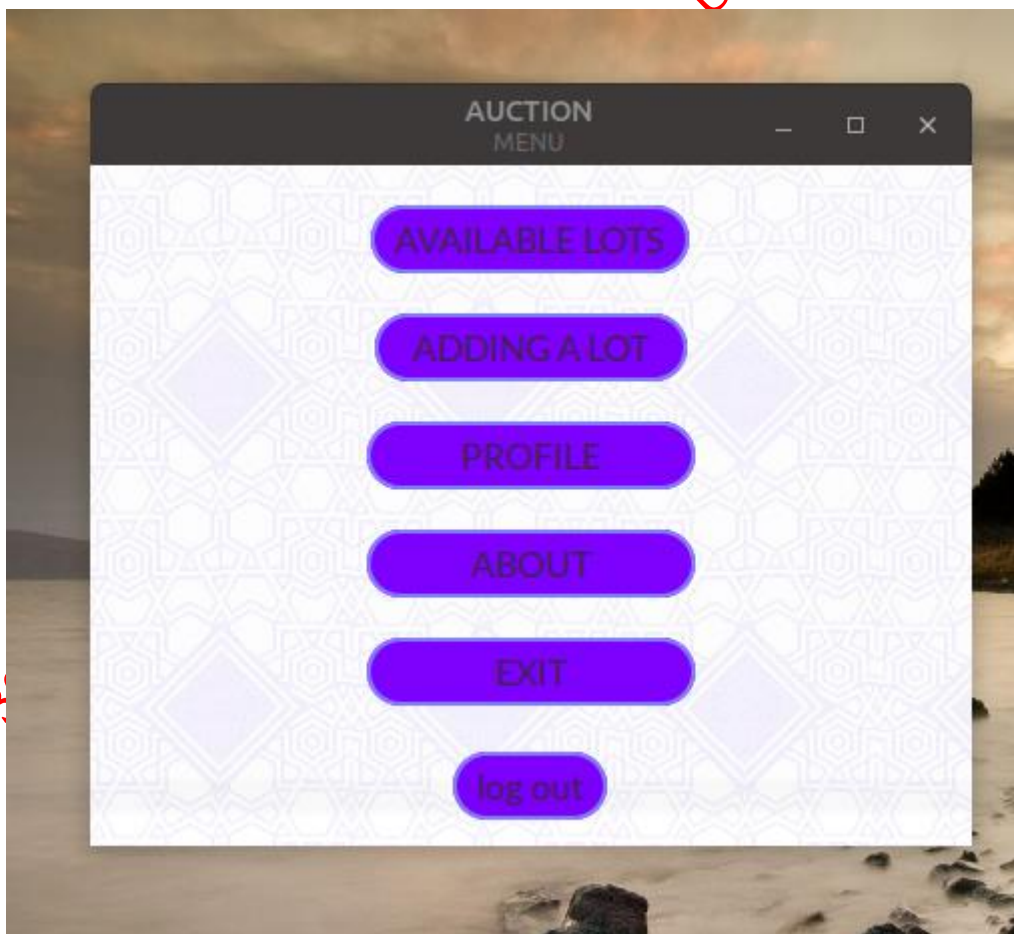
Here the main features of the application are illustrated in a one single application menu. This main menu does connect all other parts with one other all the main components and the functionalities of the application. This part is understood as a main part of the application also because it does control the operations around it with including server side and client side parts. The database system of the application is



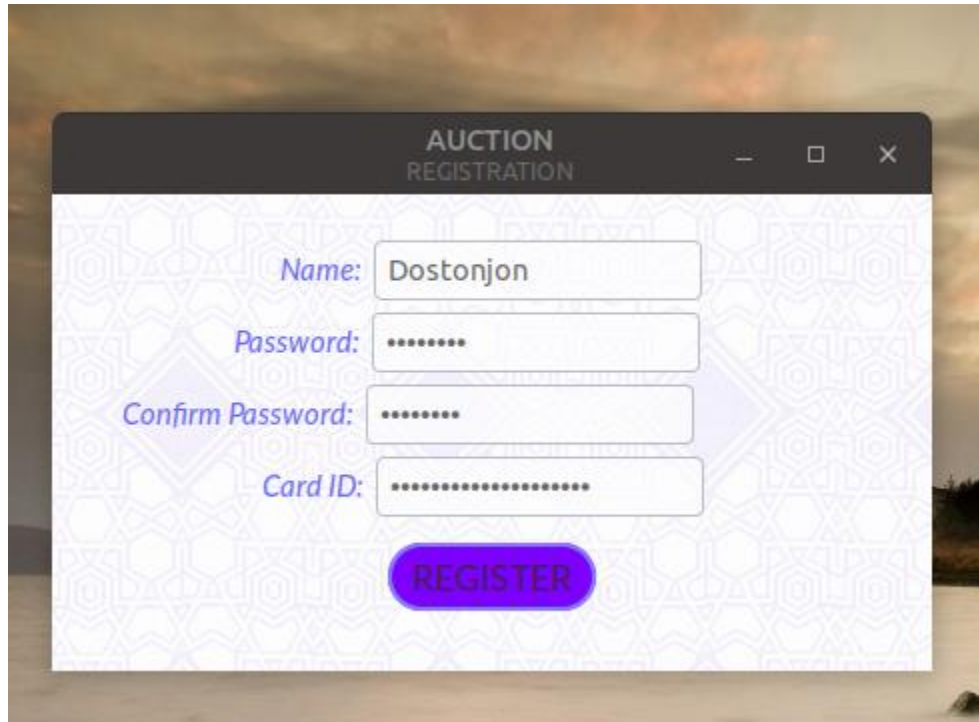
The image above is the database of the application which is already connected with a system and ready for the utilization.



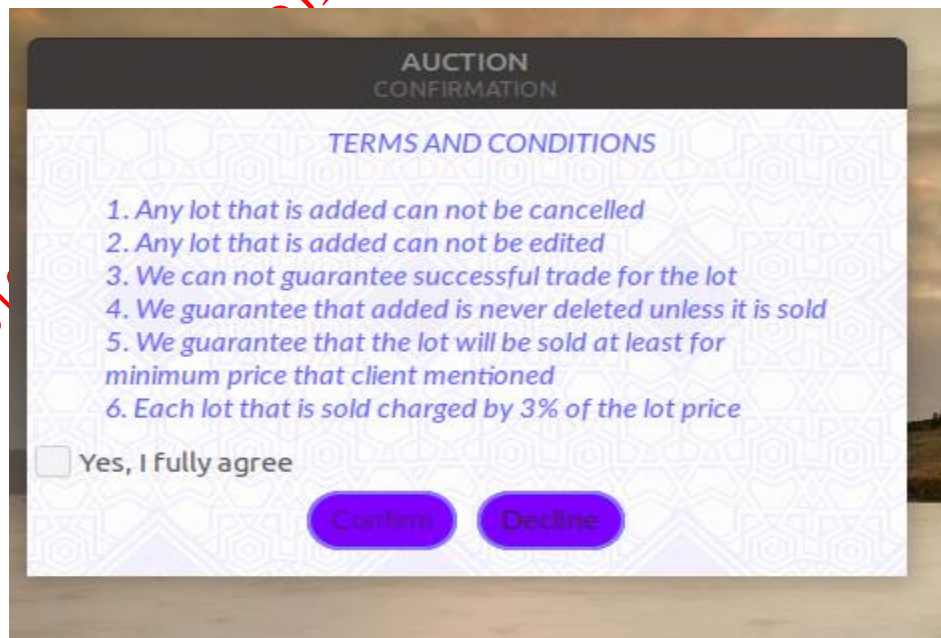
A user with a username Anvar types his login information and clicks *LOGIN* button.

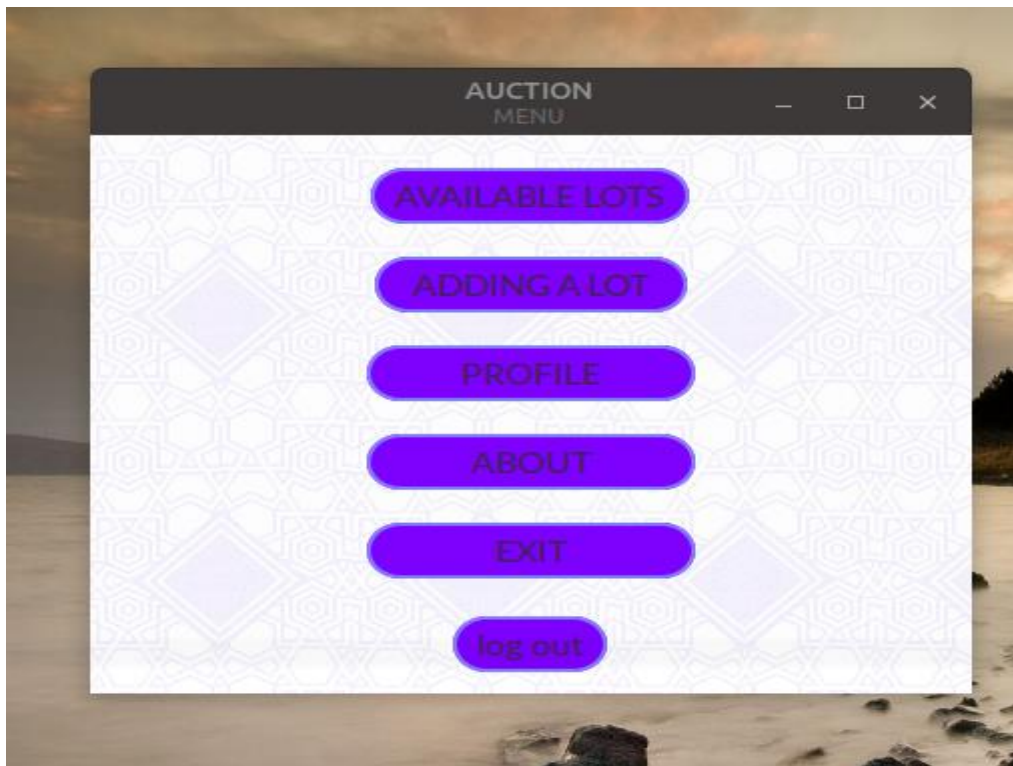


After, user is taken to the Main Menu.

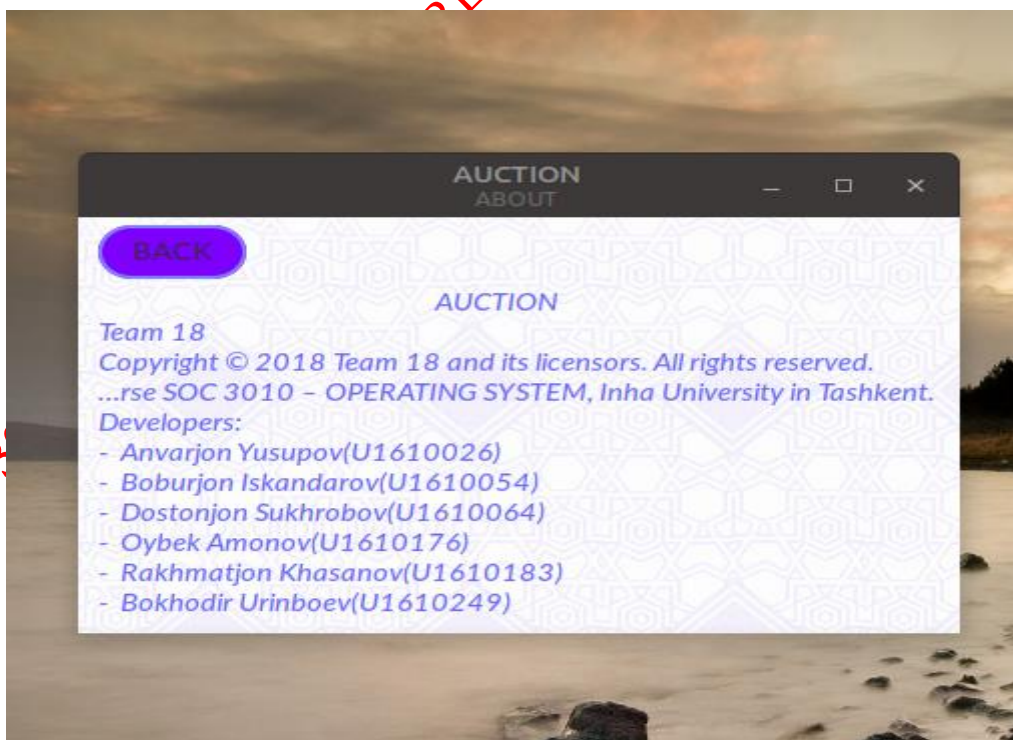


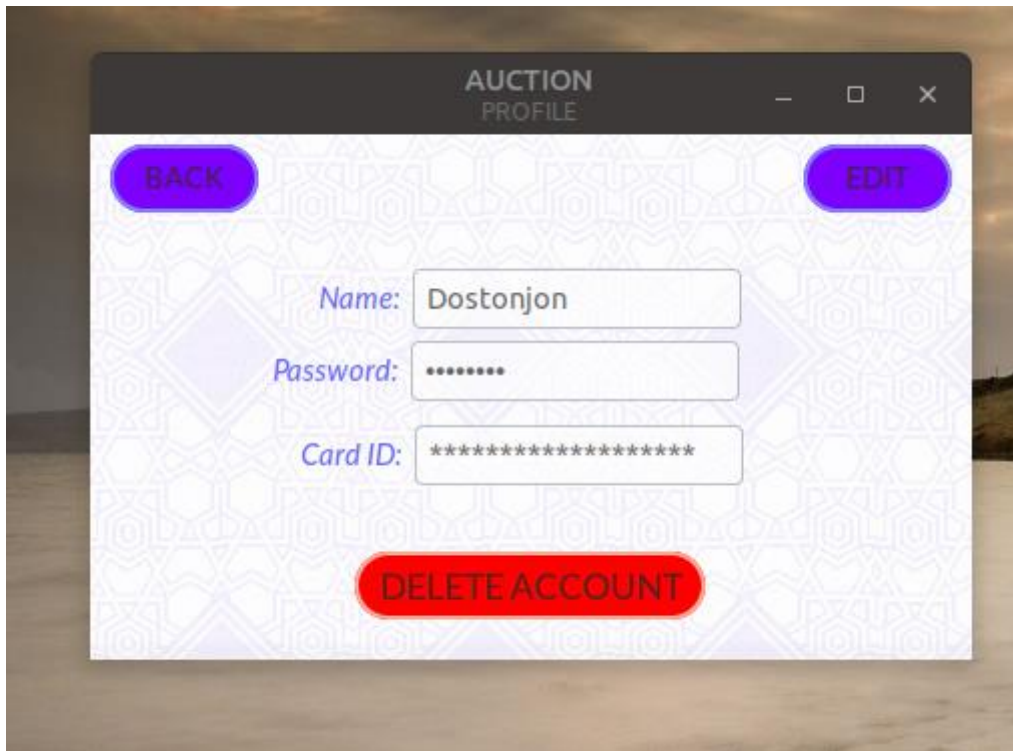
Here, a user with name Dostonjon wants to register himself. He provided the required information, and presses the REGISTER button. Afterwards, he is taken to the CONFIRMATION window, where TERMS&CONDITIONS is listed. User can either accept the terms or decline it. In case declining, user is not registered. However, if user accepts, his information is first checked for ambiguity among users and visa card validity. Above information was not in the database, and client is new, also he has a valid visa card. Thus, he is successfully registered and taken to the Main Menu.



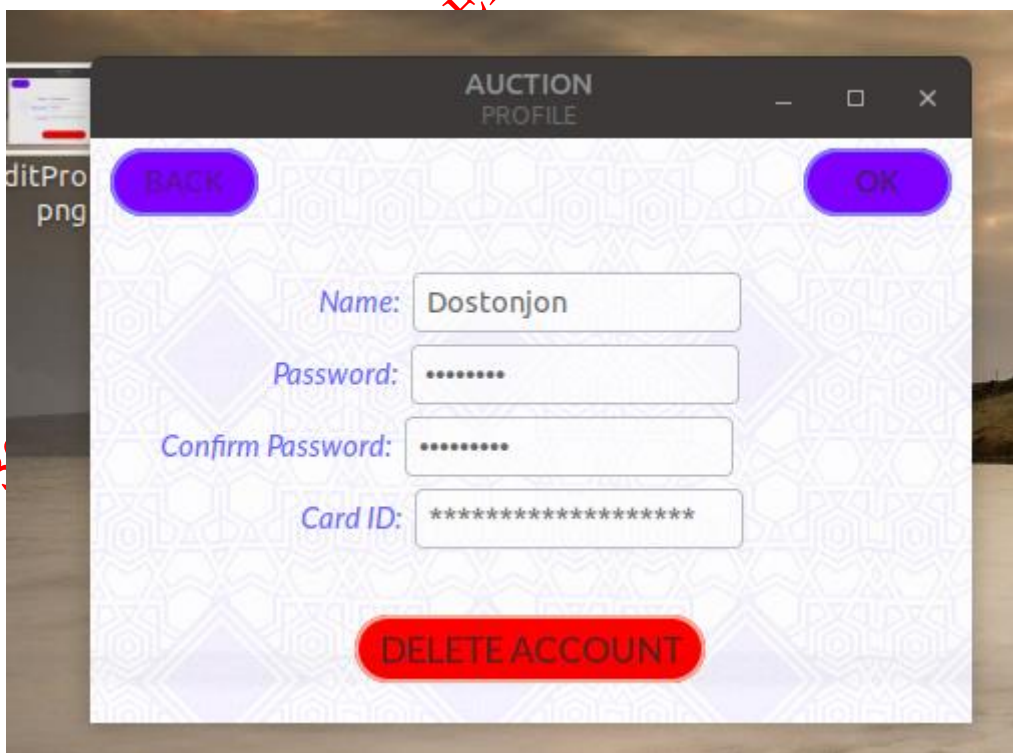


Now, user is in the main menu. And, he chooses the About button and wants to the information about the application. Below is the About window where all information is present. Copyright information and developers' names are presented.





Now, in the above picture, our current user is in Profile. And, he realized that his password is not complex enough. He clicks the button Edit and he is in Edit mode. He changes his information as he wishes and presses OK button as in below picture.



AUCTION
ADDING A LOT

BACK

Title: Flying Car

Min Price: 75.20

Description: Super Puper Car

ADD

Now, user wants to insert his lot for sale. He presses the ADDING A LOT button and he is in the above window where he has to type the information on the lot. You can see the fields above, when he presses ADD button. Again, he is asked to confirm the request by going through the TERMS&CONDITIONS again.

AUCTION
CONFIRMATION

TERMS AND CONDITIONS

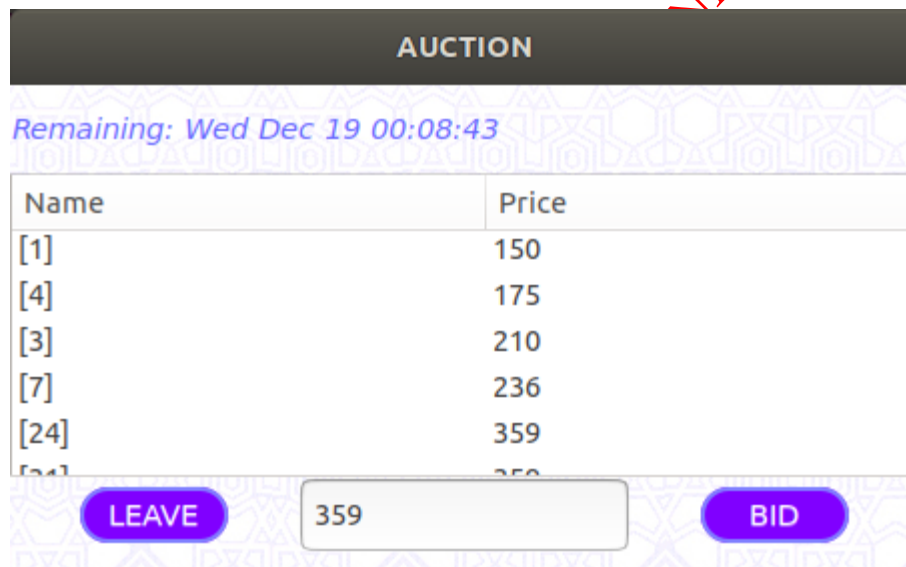
1. Any lot that is added can not be cancelled
2. Any lot that is added can not be edited
3. We can not guarantee successful trade for the lot
4. We guarantee that added is never deleted unless it is sold
5. We guarantee that the lot will be sold at least for minimum price that client mentioned
6. Each lot that is sold charged by 3% of the lot price

☒ Yes, I fully agree

Confirm Decline



Assuming that user wants to participate in an auction, he can choose the *AVAILABLE SLOTS* and the above picture will appear with all available slots. User can choose to see more on the offered lot by clicking the *MORE* button and join the auction.



The above picture depicts the actual auction bidding process. Each client connected to server is given an ID, it is in column name in brackets. And, next to the name column there is a price column where each client gives a bid. The rule for bidding is that only when a client offers a price which is greater than previous bid, the current bid is accepted otherwise it is not accepted.

CONCLUSION:

In the end of the project's team work, we have made complete working auction which does have fully functional application based functions and features on the open source operating system Linux Ubuntu and all the tools related with it additionally a combination of C language, MYSQL and CSS (for stylization and design of the application windows). With this application the users are presented with real time opportunity to coordinate with each other in the bidding system for the auction process related operations. The GUI and the client side is also fully functional with presenting the user to monitor the availability of the lots and also giving the opportunity to add their own lot for the beginning of the new auction. Additionally, all features of the modern application system which does include "Profile", "About", "Exit" and "Log Out" button for the control. The overall result is good enough for even its release as we are quite confident for its functionality and overall operational work. At the last moments, we managed to implement the hashing function into the system. Using SHA1 transformation, we improved the security issues of our system.

FUTURE WORK:

For the upcoming years our team together is going to continue developing this technology toward the high industrial standard for making it to be ready for the utilization in the open market. However, the principal issue that we have currently is that the absence of the multi-thread synchronization system in work. Also, we planned to connect the application to the existing money transaction systems like PayPal, Payme or Click. Unfortunately, we could not manage time for the realization of this functionality and therefore decided to include this part of the application for the future implementation. So, our team understood that for current application which we are presenting for the academic purposes money transaction system would be redundant functionality as all the operations could be done with virtual credit cards and all participants with fake identities. At last, we have found out that our system needs some functionality like the opportunity to see whether the lots a user inserts are sold and money transactions for that is made.

REFERENCES:

- [1]. <http://dev.mysql.com>
- [2]. https://www.w3schools.com/colors/colors_picker.asp
- [3]. <https://stackoverflow.com/questions/5141960/get-the-current-time-in-c>
- [4]. <https://stackoverflow.com/questions/8352027/gtk-timer-how-to-make-a-timer-within-a-frame>
- [5]. <https://www.geeksforgeeks.org/socket-programming-in-cc-handling-multiple-clients-on-server-without-multi-threading/>
- [6]. <https://gist.github.com/oleksiiBobko/43d33b3c25c03bcc9b2b>
- [7]. <https://www.relationaldbdesign.com/extended-database-features/module1/course-database-project.php>
- [8]. <https://developer.gnome.org/gtk3/stable/GtkListStore.html>
- [9]. <https://habr.com/post/116268/>
- [10]. <https://prognotes.net/2015/07/gtk-3-glade-c-programming-template/>
- [11]. <https://prognotes.net/2016/03/gtk-3-c-code-hello-world-tutorial-using-glade-3/>
- [12]. <https://ubuntuforums.org/archive/index.php/t-1176046.html>
- [13]. <https://developer.gnome.org/gtk3/stable/TreeWidget.html>

PROJECT TEAM:

Team Leader:

<u>Name</u>	<u>Student ID</u>	<u>Signature with date</u>
1. Bokhodir Urinboev	U1610249	19.12.2018 _____

Team Members:

<u>Name</u>	<u>Student ID</u>	<u>Signature with date</u>
2. Anvarjon Yusupov	U1610026	19.12.2018 _____
3. Boburjon Iskandarov	U1610054	19.12.2018 _____
4. Dostonjon Sukhrobov	U1610064	19.12.2018 _____
5. Oybek Amonov	U1610176	19.12.2018 _____
6. Rakhmatjon Khasanov	U1610183	19.12.2018 _____

SOC 3010-OS TERM PROJECT REPORT INHA UNIVERSITY FALL 2018