

TOPIC: ONLINE AUCTION WEBSITE DATABASE

SR NAVYA SREE

16BCE0223

RAJDEEPA CHAKRABARTY

16BCE0732

Prepared for

DATABASE MANAGMENT SYSTEMS(CSE2004)-

PROJECT COMPONENT

Submitted To

GEETHA MARY A

School of Computer Science and Engineering



VIT[®]
UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)

School of Computing Science and Engineering

1)ABSTRACT:

Our project includes a web application that will be handy when transacting. The bidder will be able to place his bids without the track of malicious people. The user will be able to put his details which will be saved and be able to be viewed only by administrator. This will ensure the safety from being exposed to robbers.

The users will also be able to request for the bid and it may be even possible for them to keep track of other peoples bids for competition.

The admin will be able to view details of the user add auction item, view progress as the users continue to bid, update and delete auction items as per requirement. Users do not have to travel , they can sit in their place and participate in the auction.

2)INTRODUCTION:

The purpose of this project is to build an “on-line auction management system”, a place for buyers and sellers to come together and trade almost anything. In fact, the system consists in a web-portal where registered users can propose new auctions, place bids in order to buy the items on auction, send messages to other users and receive automatically news via e-mail (when they receive new offers for the proposed auctions, when an auction is over etc.).

Registration of users is preceded by a “pre-registration”: to check whether users insert their real e-mail address, they receive an e-mail with an auto-generated secret code that they will be asked to type in a second moment to confirm the data (name, address, phone number etc.) they entered. Without this confirmation, a user cannot access the functionality of the portal. Auctions have a name, a description, possibly a photo (of the related item) uploaded by users and an end period: users cannot place bids when the auction interval (start - end period) ends, but, in case there were no offers for an item, there is the possibility to extend the interval. Moreover, administrators have the possibility to accept or refuse auctions proposed by users, to view information about users and items and to create, modify and delete the categories of auctions (auctions regarding cars, books, music stuff etc.). The system is realized with a 3-tier architecture: a relational database that store the information regarding items, users, auctions and categories of auction; an application server that cares about the business logic of the system and the presentation layer that consists in the web browser where users can interact with the system. With such architecture, the database is never directly accessed: for example administrators can change the data stored in the database without connecting directly to it but using their own browser.

3)AIM:

The purpose of this project is to build an “on-line auction database management system”, a place for buyers and sellers to come together and trade almost anything. In fact, the system consists in a web-portal where registered users can propose new auctions, place bids in order to buy the items on auction, send messages to other users and receive automatically news via e-mail (when they receive new offers for the proposed auctions, when an auction is over etc.).

4) MOTIVATION:

Conventional auction systems are crowded and noisy where users have to sit and bid which causes frustration. So we want to create any online auction system so that the bidding process can be conducted on a global scale.

5)OBJECTIVES:

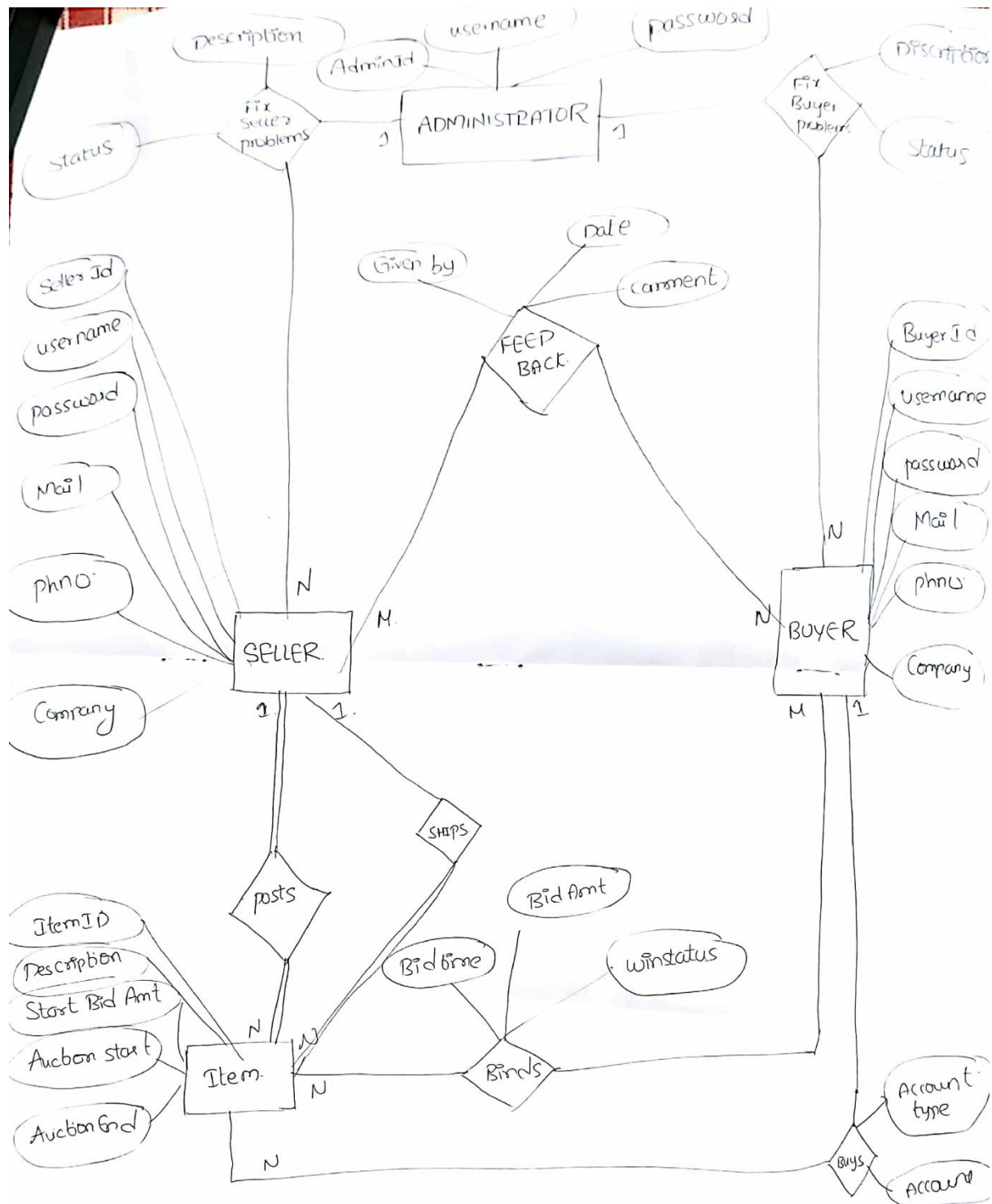
1) to create er diagram for an online auction system

2) to form a relational schema for the above mentioned problem statement.

3) challenges:

- Create an MySql database for an online auction website including the following:
- Narrative of system specifications and design
- Entity-relationship diagram
- Schema mapping diagram
- Data dictionary
- SQL used to create all tables
- queries that demonstrate how users might search for data

6) ER DIAGRAM



7) RELATIONAL TABLE

ADMINISTRATOR

ADMINID	USERNAME	PASSWORD
---------	----------	----------

SELLER

SELLER ID	USER NAME	PASSWORD	MAIL	PHONE NUMBER	COMPANY	DISCRIPTION	STATUS	ADMIN ID
-----------	-----------	----------	------	--------------	---------	-------------	--------	----------

FEEDBACK

SELLER ID	GIVEN BY	DATE	COMMENTS	BUYER ID
-----------	----------	------	----------	----------

BUYER

BUYER ID	USER NAME	PASSWORD	MAIL	PHONE NUMBER	COMPANY	DISCRIPTION	STATUS	ADMIN ID
----------	-----------	----------	------	--------------	---------	-------------	--------	----------

ITEM

ITEM ID	BUYER ID	SELLER ID	DISCRIPTION	START BID AMOUNT	AUCTION START	AUCTION END	ACCOUNT TYPE	ACCOUNT
---------	----------	-----------	-------------	------------------	---------------	-------------	--------------	---------

BIDS

BID TIME	BID AMOUT	WIN STATUS	SELLER ID	BUYER ID
----------	-----------	------------	-----------	----------

8) SOFTWARE REQUIREMENTS:

1. Database layer: my sql
2. Server side scripting : java swing

9) THREE-TIER ARCHITECTURE

For the realization of the on line auction system we used a 3-tier system architecture as shown on this schema.

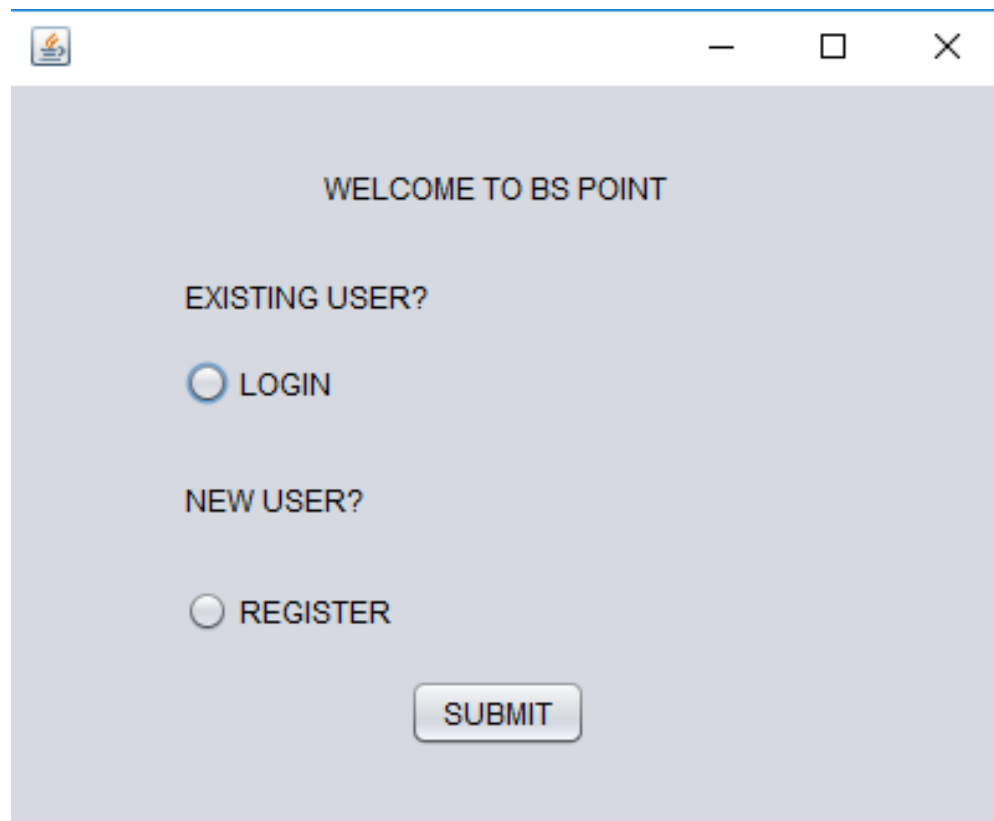
In such architecture, there are 3 main elements:

- The client tier, that is responsible for the presentation of data, receiving user elements and controlling the user interface.
- The application server tier, that is responsible for the business logic of the system. In fact, business-objects that implement the business rules "live" here, and are available to the client-tier. This tier protects the data from direct access by the clients. For the project, we used JBoss as application server.
- The data server tier, that is responsible for data storage. As data server, we used MYSQL, an open-source relational database.

10) OUTCOMES:

- Details of administrators, sellers and buyers ,items added by users are contained within the database.
- Preferred bidders are all contained within a single database
- Easy comparision of bids to decide the winner of an item.
- Better efficiency
- Easy and efficient front end

SCREEN SHOTS OF IMPLEMENTATION



WELCOME TO BS POINT

EXISTING USER?

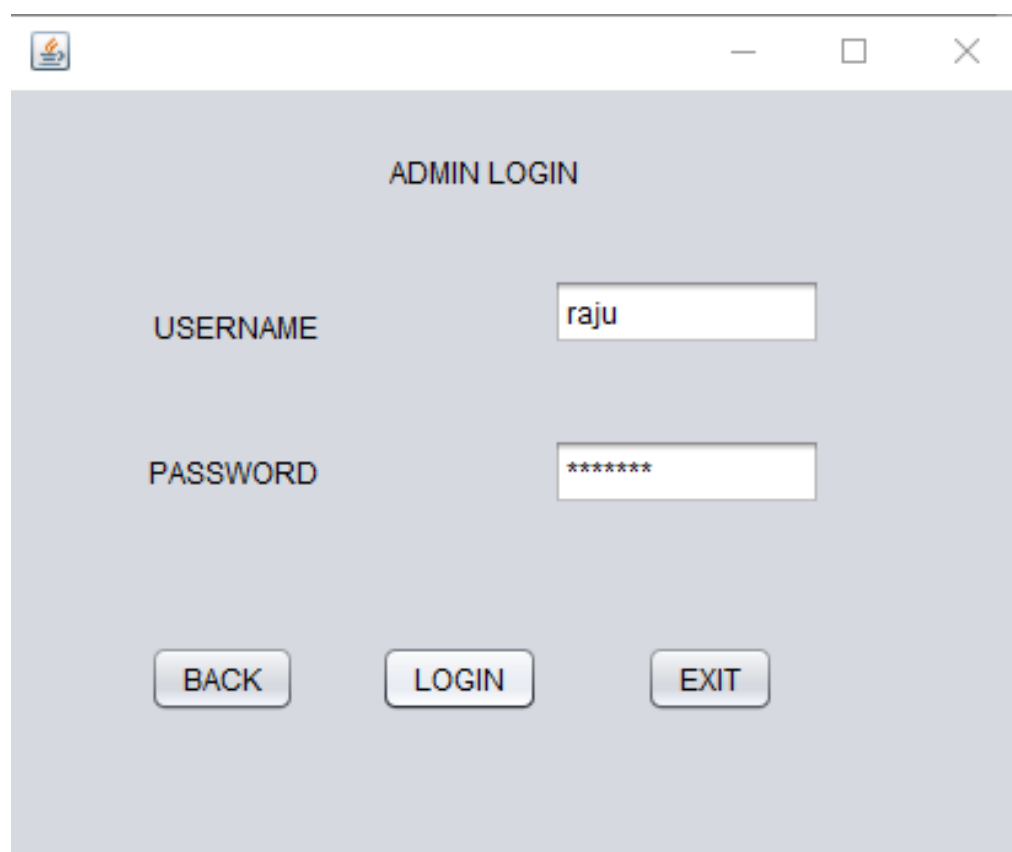
☐ LOGIN

NEW USER?

☐ REGISTER

SUBMIT

This is a Java Swing window with a light gray background. It features a title bar with a standard icon, a minus button, a maximize button, and a close button. The main content area contains the text 'WELCOME TO BS POINT' at the top. Below it, there are two sections: 'EXISTING USER?' with a radio button labeled 'LOGIN', and 'NEW USER?' with a radio button labeled 'REGISTER'. At the bottom center, there is a button labeled 'SUBMIT'.



ADMIN LOGIN

USERNAME raju

PASSWORD *****

BACK LOGIN EXIT

This is a Java Swing window with a light gray background. It features a title bar with a standard icon, a minus button, a maximize button, and a close button. The main content area contains the text 'ADMIN LOGIN' at the top. Below it, there are two input fields: 'USERNAME' with the text 'raju' and 'PASSWORD' with masked text '*****'. At the bottom, there are three buttons: 'BACK', 'LOGIN', and 'EXIT'.

Message



Login Successful

OK

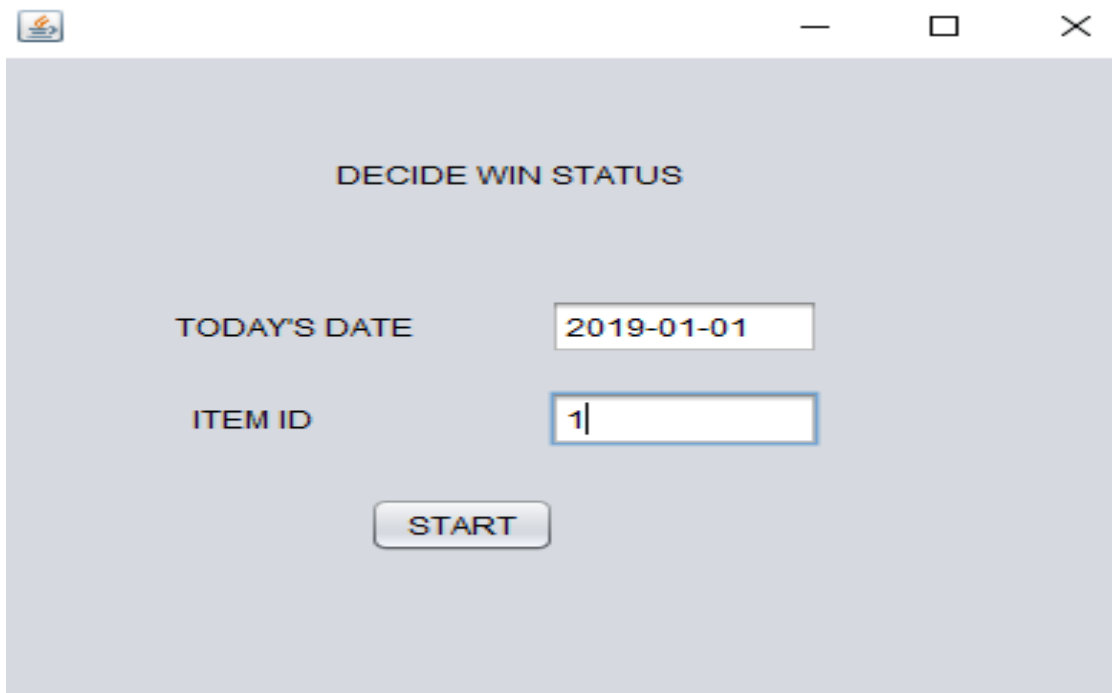


WHAT TASK DO YOU WANT TO EXECUTE

☒ SEE TO USER/BUYER PROBLEMS

☐ EVALUATE WIN STATUS OF PRODUCTS

SUBMIT

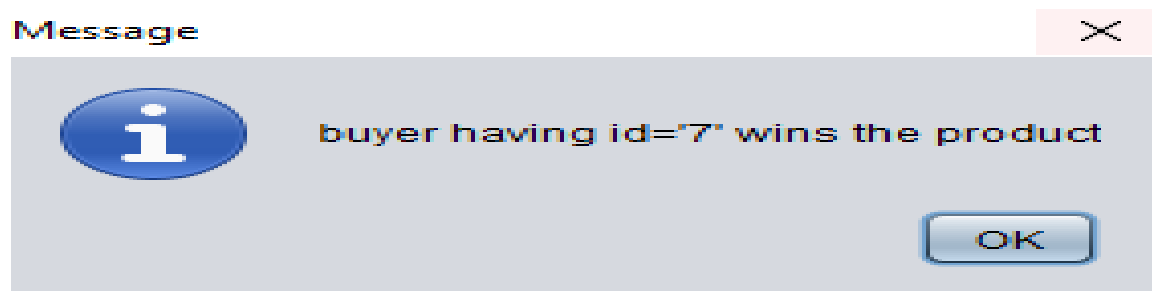


DECIDE WIN STATUS

TODAY'S DATE 2019-01-01

ITEM ID 1

START



LOGIN PAGE SAMPLE CODE

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;

public class Buyer_Login extends javax.swing.JFrame {

    /**
     * Creates new form Buyer_Login
     */
}
```

```
*/

public Buyer_Login() {

    initComponents();

}

/**

 * This method is called from within the constructor to initialize the form.

 * WARNING: Do NOT modify this code. The content of this method is always

 * regenerated by the Form Editor.

 */

@SuppressWarnings("unchecked")

// <editor-fold defaultstate="collapsed" desc="Generated Code">

private void initComponents() {

    jLabel1 = new javax.swing.JLabel();

    jLabel2 = new javax.swing.JLabel();

    jLabel3 = new javax.swing.JLabel();

    un = new javax.swing.JTextField();

    pwd = new javax.swing.JPasswordField();

    log = new javax.swing.JButton();

    exit = new javax.swing.JButton();

    back = new javax.swing.JButton();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jLabel1.setText("BUYER LOGIN");
```

```
jLabel2.setText("USERNAME");
```

```
jLabel3.setText("PASSWORD");
```

```
un.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        unActionPerformed(evt);  
    }  
});
```

```
log.setText("LOGIN");  
log.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        logActionPerformed(evt);  
    }  
});
```

```
exit.setText("EXIT");  
exit.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {  
        exitActionPerformed(evt);  
    }  
});
```

```
back.setText("BACK");  
back.addActionListener(new java.awt.event.ActionListener() {  
    public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
        backActionPerformed(evt);
    }
});
```

```
javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());  
getContentPane().setLayout(layout);  
layout.setHorizontalGroup(  
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
        .addGroup(layout.createSequentialGroup()  
  
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
            .addGroup(layout.createSequentialGroup()  
  
                .addGap(63, 63, 63)  
  
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
            .addComponent(jLabel2)  
            .addComponent(jLabel3))  
            .addGap(99, 99, 99)  
  
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,  
false)  
            .addComponent(pwd, javax.swing.GroupLayout.DEFAULT_SIZE, 101,  
Short.MAX_VALUE)  
            .addComponent(un)))  
        .addGroup(layout.createSequentialGroup()  
  
            .addGap(147, 147, 147)  
  
            .addComponent(jLabel1)))  
    .addContainerGap(80, Short.MAX_VALUE))  
    .addGroup(layout.createSequentialGroup()
```

```

        .addGap(39, 39, 39)

        .addComponent(back)

        .addGap(45, 45, 45)

        .addComponent(log)

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 97,
Short.MAX_VALUE)

        .addComponent(exit)

        .addGap(42, 42, 42))

    );

    layout.setVerticalGroup(

        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

        .addGroup(layout.createSequentialGroup()

            .addGap(24, 24, 24)

            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

                .addComponent(pwd, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

                .addGroup(layout.createSequentialGroup()

                    .addComponent(jLabel1)

                    .addGap(53, 53, 53)

                    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)

                        .addComponent(jLabel2)

                        .addComponent(un, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

                        .addGap(54, 54, 54)

                        .addComponent(jLabel3)))
            )
        )
    )
}

```

```
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 45,
Short.MAX_VALUE)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
```

```
    .addComponent(log)
```

```
    .addComponent(exit)
```

```
    .addComponent(back))
```

```
    .addGap(53, 53, 53))
```

```
);
```

```
pack();
```

```
}// </editor-fold>
```

```
private void unActionPerformed(java.awt.event.ActionEvent evt) {
```

```
    // TODO add your handling code here:
```

```
}
```

```
private void logActionPerformed(java.awt.event.ActionEvent evt) {
```

```
    // TODO add your handling code here:
```

```
String untf = un.getText();
```

```
String pf = new String(pwd.getPassword());
```

```
try{
```

```
    Class.forName("java.sql.Driver");
```

```
    Connection conn =
```

```
DriverManager.getConnection("jdbc:mysql://localhost:3306/auction_server", "root",
"rumrum");
```

```
    Statement stmt = conn.createStatement();
```

```
    String sql = "select *from tbl_buyer where bUserName = '" + untf + "'";
```

```

        ResultSet rs = stmt.executeQuery(sql);

        rs.next();

        String str = rs.getString("bPassword");

        // String a = rs.getString("Name");

        if(str.equals(pf))
        {
            JOptionPane.showMessageDialog(null, "Login Successful");

            Buyer_before_view l = new Buyer_before_view();
            l.setVisible(true);
            this.setVisible(false);
        }
        else
        {
            JOptionPane.showMessageDialog(null, "Incorrect username or password");
        }
    } catch (Exception e)
    {
        JOptionPane.showMessageDialog(null, "Incorrect username or password");
    }
}

private void exitActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    this.setVisible(false);
}

private void backActionPerformed(java.awt.event.ActionEvent evt) {

```

```

// TODO add your handling code here:

Login l=new Login();

l.setVisible(true);

this.setVisible(false);

}

/**
 * @param args the command line arguments
 */

public static void main(String args[]) {

    /* Set the Nimbus look and feel */

    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">

    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and
feel.

        * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */

    try {

        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {

            if ("Nimbus".equals(info.getName())) {

                javax.swing.UIManager.setLookAndFeel(info.getClassName());

                break;

            }

        }

    } catch (ClassNotFoundException ex) {

        java.util.logging.Logger.getLogger(Buyer_Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
    }
}

```



```

        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Buyer_Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Buyer_Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Buyer_Login.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

        }
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
    public void run() {
        new Buyer_Login().setVisible(true);
    }
});
}

// Variables declaration - do not modify
private javax.swing.JButton back;
private javax.swing.JButton exit;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;

```

```

private javax.swing.JButton log;

private javax.swing.JPasswordField pwd;

private javax.swing.JTextField un;

// End of variables declaration
}

```

BACKEND

```
mysql> use auction_server;
```

Database changed

```
mysql> desc tbl_administrator;
```

Field	Type	Null	Key	Default	Extra
adminId	int(11)	NO	PRI	NULL	auto_increment
aUserName	varchar(45)	NO	UNI	NULL	
aPassword	varchar(45)	NO		NULL	

3 rows in set (0.02 sec)

```
mysql> desc tbl_seller;
```

Field	Type	Null	Key	Default	Extra
sellerId	int(11)	NO	PRI	NULL	auto_increment
sUserName	varchar(45)	NO	UNI	NULL	
sPassword	varchar(45)	NO		NULL	
company	varchar(45)	NO		NULL	

smailId	varchar(45)	YES		NULL		
sproblemStatus	varchar(45)	YES		NULL		
sproblemDescription	varchar(45)	YES		NULL		
sphonenumner	int(10)	NO		NULL		
adminId	int(11)	NO	MUL	NULL		

+-----+-----+-----+-----+-----+-----+

9 rows in set (0.08 sec)

mysql> desc tbl_buyer;

Field	Type	Null	Key	Default	Extra	
-------	------	------	-----	---------	-------	--

+-----+-----+-----+-----+-----+-----+

buyerId	int(11)	NO	PRI	NULL	auto_increment	
bUserName	varchar(45)	NO	UNI	NULL		
bPassword	varchar(45)	NO		NULL		
address	varchar(45)	NO		NULL		
bmailId	varchar(45)	YES		NULL		
bproblemStatus	varchar(45)	YES		UNSOLVED		
bproblemDescription	varchar(45)	YES		NULL		
bphonenumner	int(10)	NO		NULL		
adminId1	int(11)	NO	MUL	NULL		

+-----+-----+-----+-----+-----+-----+

9 rows in set (0.05 sec)

mysql> desc tbl_item;

Field	Type	Null	Key	Default	Extra	
-------	------	------	-----	---------	-------	--

```

+-----+-----+-----+-----+-----+
| item_id      | int(11)  | NO   | PRI | NULL | auto_increment |
| item_desc    | varchar(45) | YES  |     | NULL |                |
| item_startbidamt | int(11)  | YES  |     | NULL |                |
| auction_startdate | date     | YES  |     | NULL |                |
| auction_enddate  | date     | YES  |     | NULL |                |
| ship_price     | int(11)  | YES  |     | NULL |                |
| ship_date      | date     | YES  |     | NULL |                |
| arrival_date    | date     | YES  |     | NULL |                |
| bacc_type      | varchar(45) | YES  |     | NULL |                |
| bacc_name      | varchar(45) | YES  |     | NULL |                |
| sellerId      | int(11)  | YES  | MUL | NULL |                |
| buyerId      | int(11)  | YES  | MUL | NULL |                |
+-----+-----+-----+-----+-----+

```

12 rows in set (0.02 sec)

```
mysql> desc tbl_bids;
```

```

+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| buyerId1  | int(11)   | YES  | MUL | NULL    |       |
| itemId1    | int(11)   | YES  | MUL | NULL    |       |
| bid_time   | date      | YES  |     | NULL    |       |
| bid_amt    | int(11)   | YES  |     | NULL    |       |
| win_status | varchar(45) | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+

```

5 rows in set (0.02 sec)

```
mysql> desc tbl_feedback;
```

ERROR 1146 (42S02): Table 'auction_server.tbl_feedback' doesn't exist

```
mysql> desc feedback;
```

```
+-----+-----+-----+-----+-----+
| Field      | Type      | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| description | varchar(45) | YES  |     | NULL    |       |
| seller_id   | int(11)    | YES  | MUL | NULL    |       |
| buyer_id   | int(11)    | YES  | MUL | NULL    |       |
+-----+-----+-----+-----+-----+
```

3 rows in set (0.05 sec)

```
mysql>
```

11) CONCLUSION:

The project report entitled “online auction” has come to its conclusion the new system has been developed with so much care that it is free of errors and at the same time efficient and less time consuming .The sellers can add items, buyers can state bids and the wining status is taken care by administrator. The sellers and buyers can communicate among themselves and also convey their problems to the administrator. System is robust also provision is provided for future developments in the system.

12)REFERENCES

- 1)fred barewell,Richard blair....(2004)' professional VB.NET 2nd edition',wrox press ltd
- 2)www.auction.indiatimes.com
- 3)[www.eba\\$.com](http://www.eba$.com)
- 4)www.msdn.com
- 5) Agile Modelling Website www.agilemodeling.com
- 6) Eclipse Website www.eclipse.org
- 7) Elmasri Ramez, Shamkant Navathe, Fundamentals of Database Systems, 2004 Addison-Wesley
- 8)Developer.com Website www.developer.com
- 9) InternetWeek Website www.internetweek.com

Thank you