
JJD Systems

System Design Document
Electronic Bidding System
CCNY Software Engineering
Professor J. Wei
Joel Kemp
Jamal Goddard
Daniel Ranells

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Revision History

Date	Version	Description	Author
4/6/06	1.0	All Sections	TEAM
5/17/06	2.0	All Sections	TEAM

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Table of Contents

1. System Class Diagram	4
E/R Model:	4
2. Interaction Diagrams	5
POST NEW ITEM INTERFACE	5
LOGIN INTERFACE	7
MY ACCOUNT INTERFACE	9
MAIN INTERFACE	11
REGISTRATION INTERFACE	13
TRANSACTION HISTORY INTERFACE	14
BROWSE INTERFACE	15
COMPLAINT	17
WITHDRAWL	19
SA Authorize	20
SA Authorize Item	21
SA Complaints	22
SA System Information	23
3. Detailed Design	24
Post Item Interface	24
Attributes	24
Functions	25
Attributes	27
Functions	28
Browse Interface	30
Attributes	30
Functions	30
Detailed Item Interface	32
Attributes	32
Functions	33
Authorization Interface	34
Attributes	34
Functions	34
Login Interface	35
Attributes	35
Functions	35
TheMain Interface	37
Attributes	37
Functions	37
Register Interface	38
Attributes	38

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Functions.....	39
Complaint Interface	41
Attributes.....	41
Functions.....	42
Transaction History Interface	44
Attributes.....	44
Functions.....	44
User Class	45
Variables:	45
Functions:.....	46
Item Class.....	48
Variables:	48
Functions:.....	49
Complaint Class	50
Variables:	50
Functions:.....	50
Database Class	52
Attributes:	52
Functions:.....	53
Transaction Class	55
Attributes:	55
Functions:.....	55
ItemWatcher.....	56
Attributes:	56
Functions:.....	56
JTextFocusHandler	57
Attributes:	57
Functions:.....	57
MyTableModel	58
Attributes:	58
Functions:.....	58
SystemStatsInterface.....	59
Attributes:	59
Functions:.....	60
TabbedPane.....	62
Attributes:	62
Functions:.....	62
4. System Screens	63

Design Document Specification

1. System Class Diagram

E/R Model:

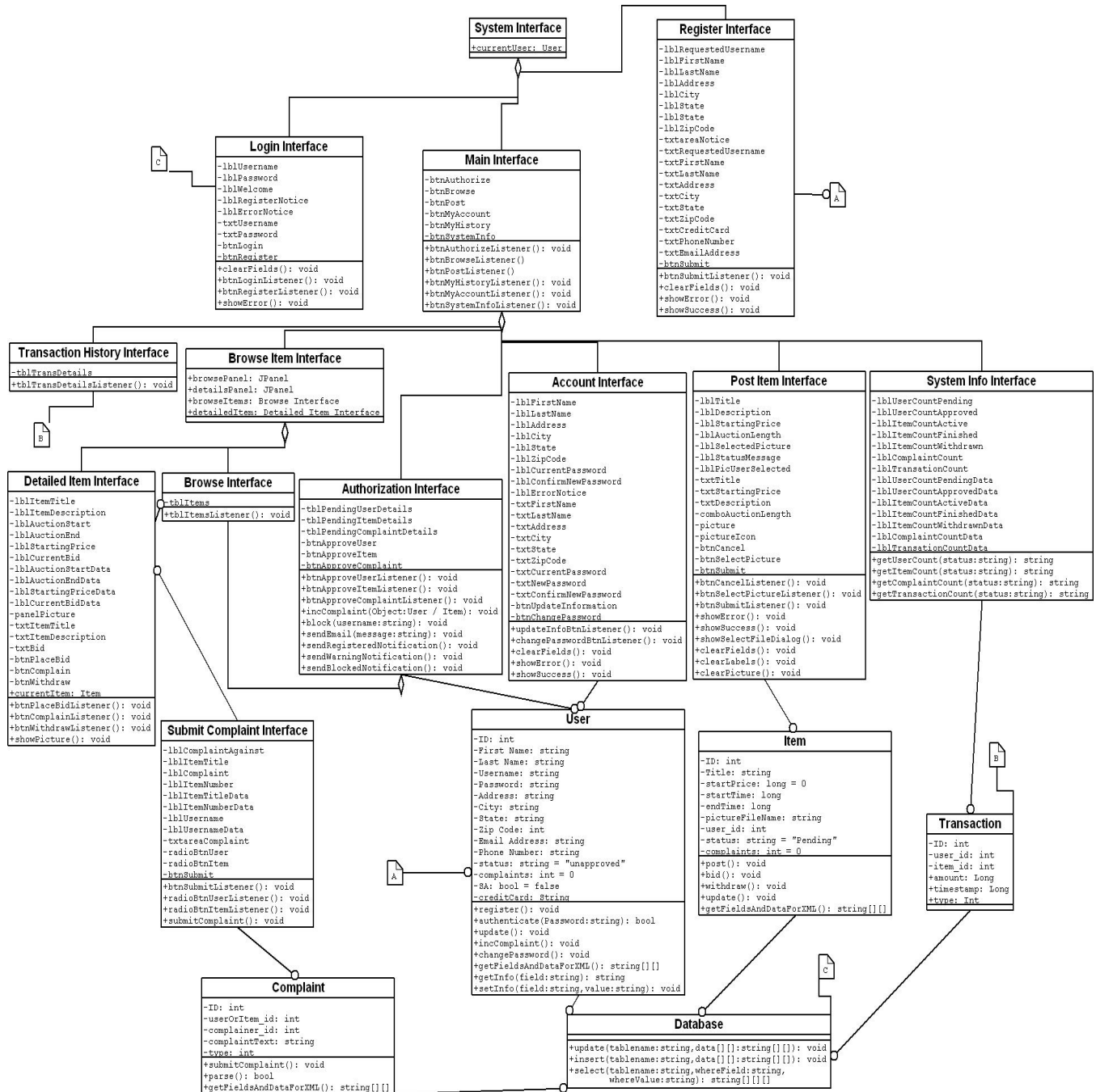


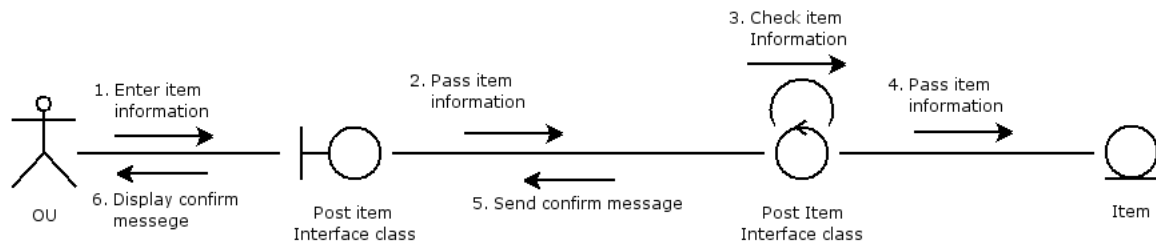
Figure 1.1 E/R Class Diagram for System

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

2. Interaction Diagrams

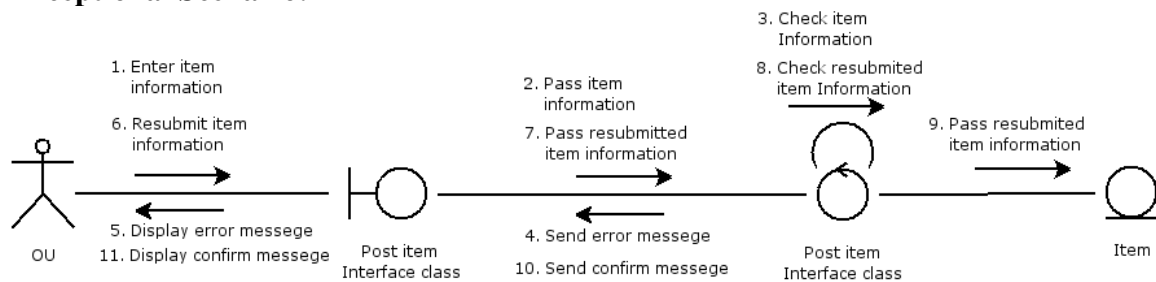
POST NEW ITEM INTERFACE

Normal Scenario:



1. OU fills in the title, description, starting bid, auction length and selects the path of picture if one is selected.
2. The OU clicks submit. The system then checks the fields for valid input and checks for the picture at the given path (Only if a picture is included).
3. Once no errors are found. The interface then displays the message “Item submitted successfully. You will receive an email when your auction has been approved by the system administrator.”
4. The system then appends this information to the item file.

Exceptional Scenario:



1. The OU inputs invalid information.
2. The OU clicks submit and the information is sent to the system.
3. The system checks the fields and finds that one of the fields is empty.
4. The system sends an error to the screen and highlights the incorrect fields.
5. The OU fixes the problem and resubmits the information to the system.
6. The system then checks the fields to make sure that everything is filled out and checks for the picture at the given path (Only if a picture is included) .

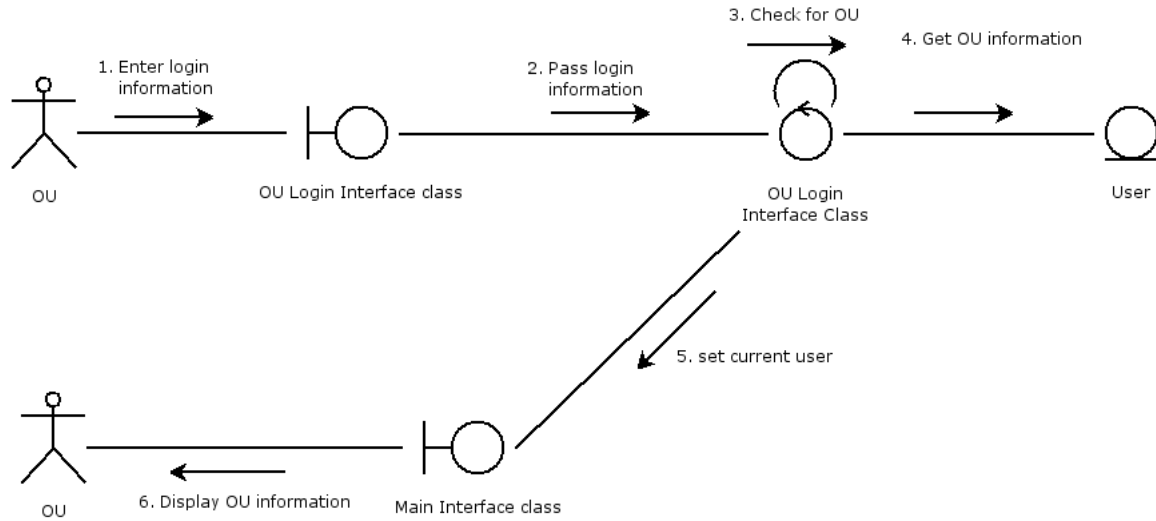
Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

7. Once no errors are found. It then displays the message “Auction submitted successfully. You will receive an email when your auction has been approved by the system administrator.”
8. The system then appends this information to an item xml file.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

LOGIN INTERFACE

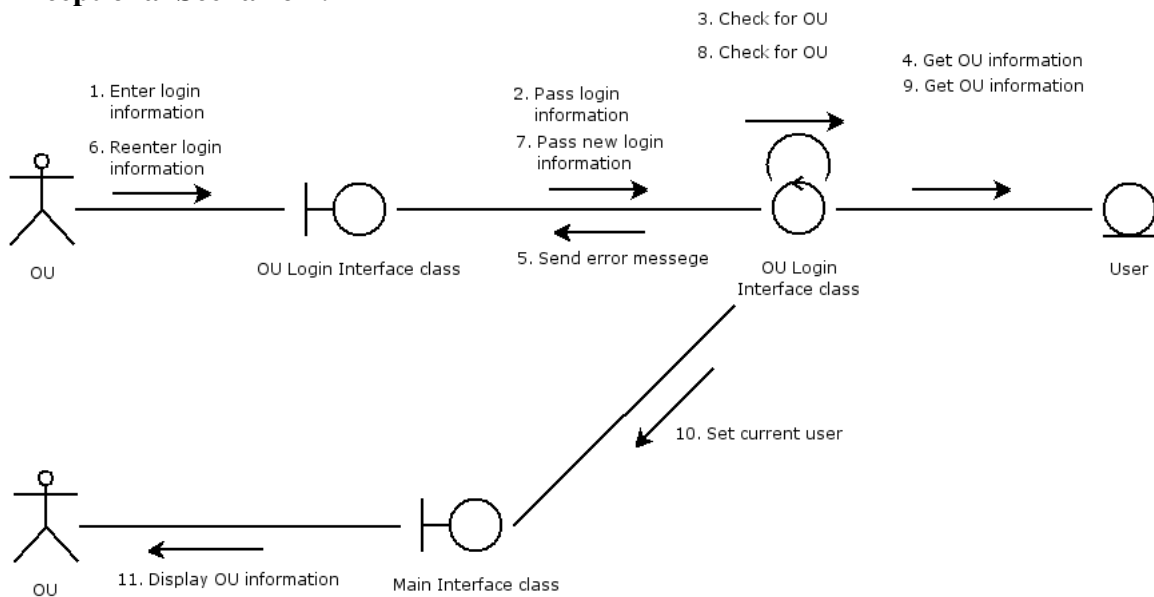
Normal Scenario:



1. OU enters their OU name and password.
2. The OU then clicks login and submits the information
3. The system then checks the OU name and password against the current registered OUs.
4. The system brings up the Main Interface

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Exceptional Scenario 1:



1. OU enter their name or password incorrectly
2. The system checks the OU user name and password against all registered OU names and passwords and doesn't find a match
3. An error is sent to the screen stating that the "OU user name and or Password doesn't exist"
4. The OU then enters the correct OU name and passwords and clicks submit.
5. The system then checks the OU name and password against the current registered OUs.
6. The system brings up the Main Interface

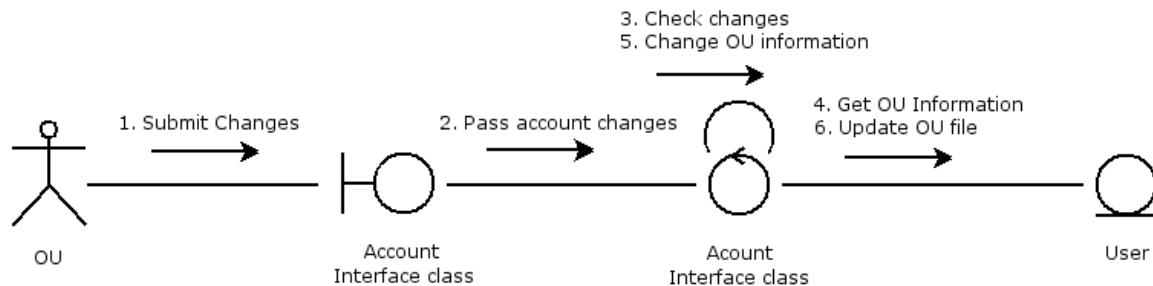
Exceptional Scenario 2:

1. A new OU is interested in using the auction system clicks on the register button.
2. The Register Interface is shown.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

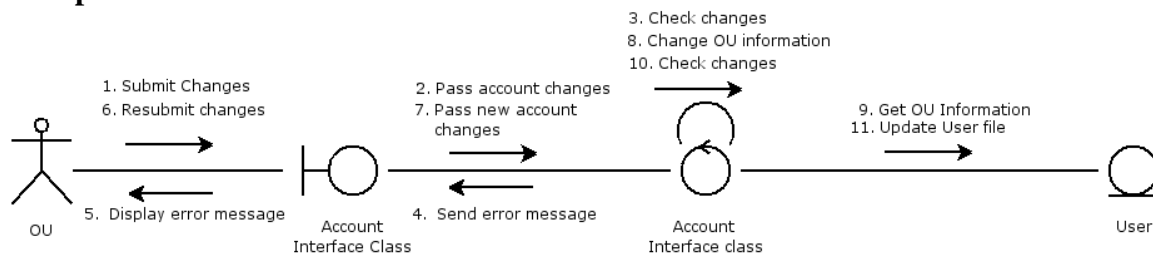
MY ACCOUNT INTERFACE

Normal Scenario:



1. An OU wants to change their current information stored on the system.
2. The OU then clicks on My Account from the Main Interface.
3. The First and Last name fields are not accessible so they remain grey once the interface is shown.
4. The address, city, state, zip code, phone#, and email address fields are all accessible.
5. The OU changes information in the accessible fields and presses the Change User Information button.
6. The fields are checked for correct data format.
7. The OU information is then changed and store on the OUs file.

Exceptional Scenario 1:



1. An OU wants to change their information
2. The OU then clicks on My Account from the Main Interface.
3. The First and Last name fields are not accessible so they remain grey once the interface is shown.
4. The address, city, state, zip code, phone#, and email address fields are all accessible.
5. The OU enters invalid information into a field.
6. The OU clicks submit

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

7. The system checks the fields and finds that a field is incorrect.
8. The incorrect field is highlight and an error is shown.
9. The OU then corrects the information and clicks the submit button.
10. The system checks the fields and finds no errors.
11. The system then changes the OUs information to the new information and stores it in the user file.

Exceptional Scenario 2:

1. The OU clicks the Change Password button and enters a new password twice.
2. The system then checks the new password fields.
3. Everything is fine and the new password is submitted to the system
4. The system then changes the OUs password to the new password and stores on the OUs file.

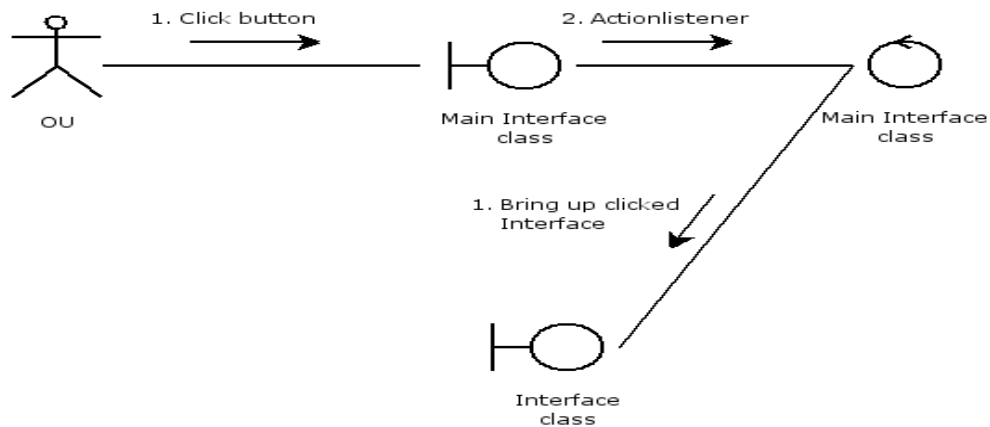
Exceptional Scenario 3:

1. The OU enters their password in the password fields incorrectly.
2. The OU clicks change password.
3. The system checks the password and finds they do not match.
4. The system then displays “Your passwords do not match” and highlights the incorrect field that is causing the error
5. The OU then cancels his transaction.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

MAIN INTERFACE

Normal Scenario 1:



1. An OU logs in and the Main Interface comes up.
2. The OU then clicks the Browse button which brings up the Browse Interface.

Normal Scenario 2:

1. An OU logs in and The Main Interface comes up.
2. The OU then clicks the Sell Browser button which brings up the Sell Interface

Normal Scenario 3:

1. An OU logs and the Main Interface comes up.
2. The OU then clicks the My History button which brings up the My History interface

Normal Scenario 4:

1. An OU logs in and the Main Interface comes up.
2. The OU then clicks the My Account button which brings up the My Account interface

Normal Scenario 5:

1. An SA logs in and the Main Interface comes up.
2. The Authorize and System Info button then becomes available.
3. The SA then clicks the Authorize button which then brings up the Authorize Interface.

Normal Scenario 6:

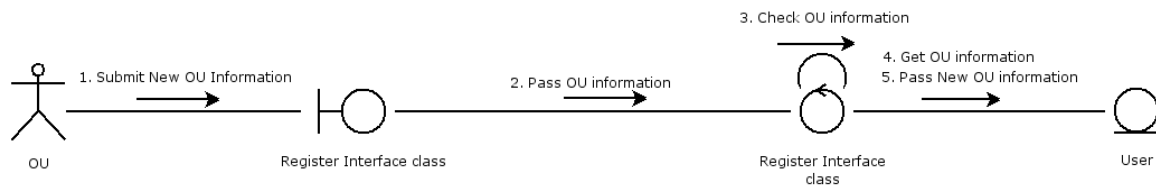
Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

1. An SA logs in and the Main Interface comes up.
2. The Authorize and System Info button then becomes available.
3. The SA then clicks the System Info button which then brings up the System Info Interface.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

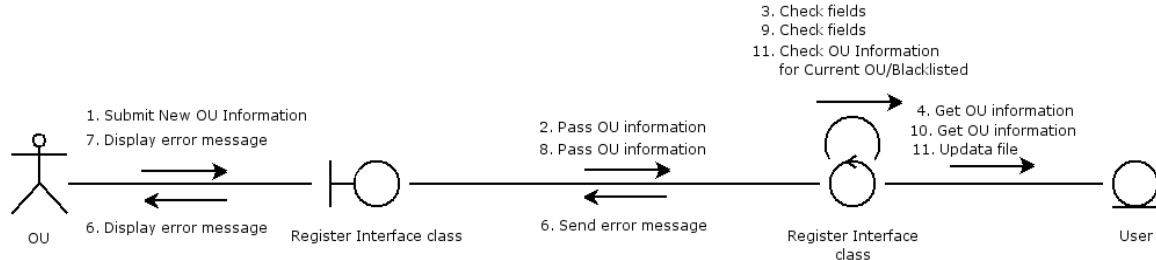
REGISTRATION INTERFACE

Normal Scenario:



1. A new OU enters their Requested OU name, First name, Last name, Address, City, State, Zip Code and then presses the submit button.
2. The system checks that all of the fields are filled in and are in the correct format.
3. The system then checks to see if the new OU is already exist.
4. If the New OU doesn't exist the system then submit the information to the new OU appending list

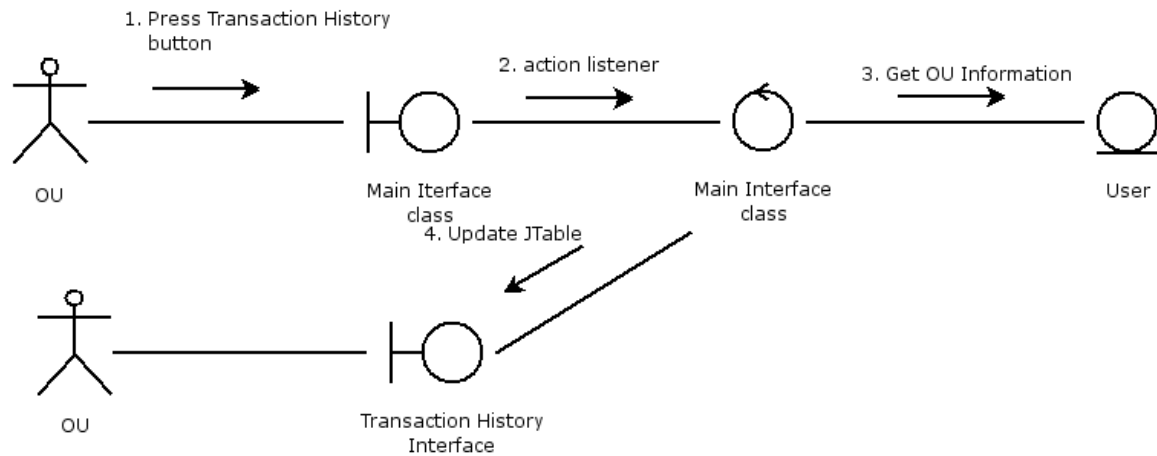
Exceptional Scenario:



1. A new OU enters their Requested OU name, First name, Last name, City, State, Zip Code and then presses the submit button.
2. The OU forgot to put in an address and clicks submit
3. The system checks the fields and finds there is no address entered
4. The system sends an error message to the interface and highlights the OU field that needs to be completed
5. The OU enters an address and clicks submit
6. The system checks that all of the fields are filled in and are in the correct format.
7. The system then checks to see if the new OU is already exist.
8. If the New OU doesn't exist the system then submit the information to the new OU appending list

TRANSACTION HISTORY INTERFACE

Normal Scenario 1:



1. The OU clicks on the transaction history button
2. The Transaction History Interface comes up with all the OU transactions

BROWSE INTERFACE

Normal Scenario

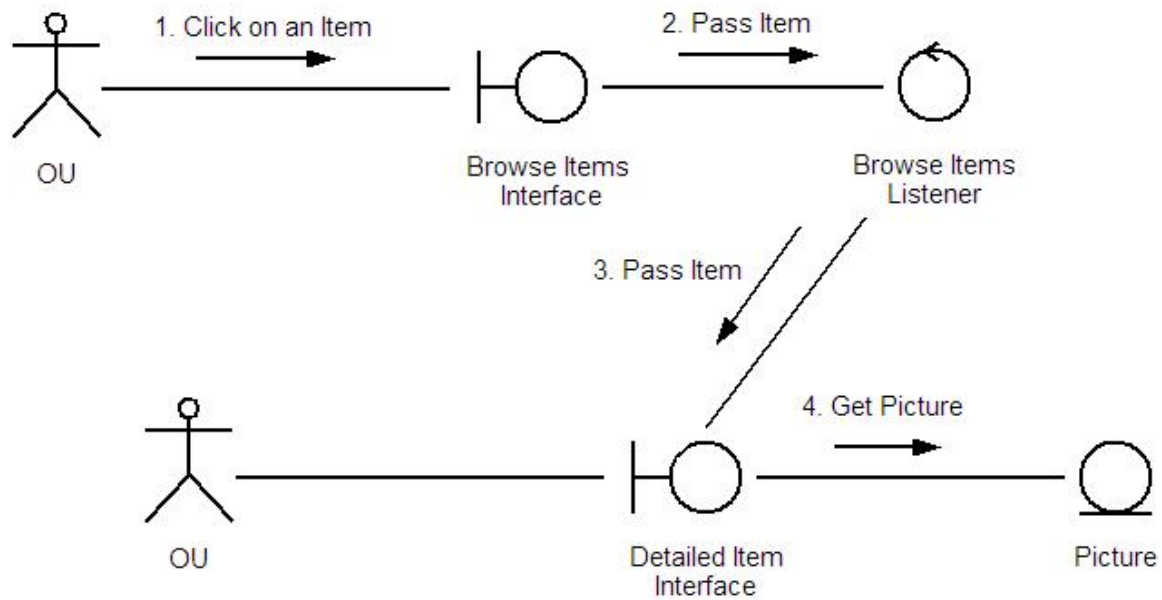


Figure 3.15 OU Browse collaboration diagram

1. The OU selects the Browse button on the Main Interface to show the Browse Item Interface.
2. The system shows a Browse Item Interface without the ability to see withdrawn items and without the button to delete items.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Normal Scenario

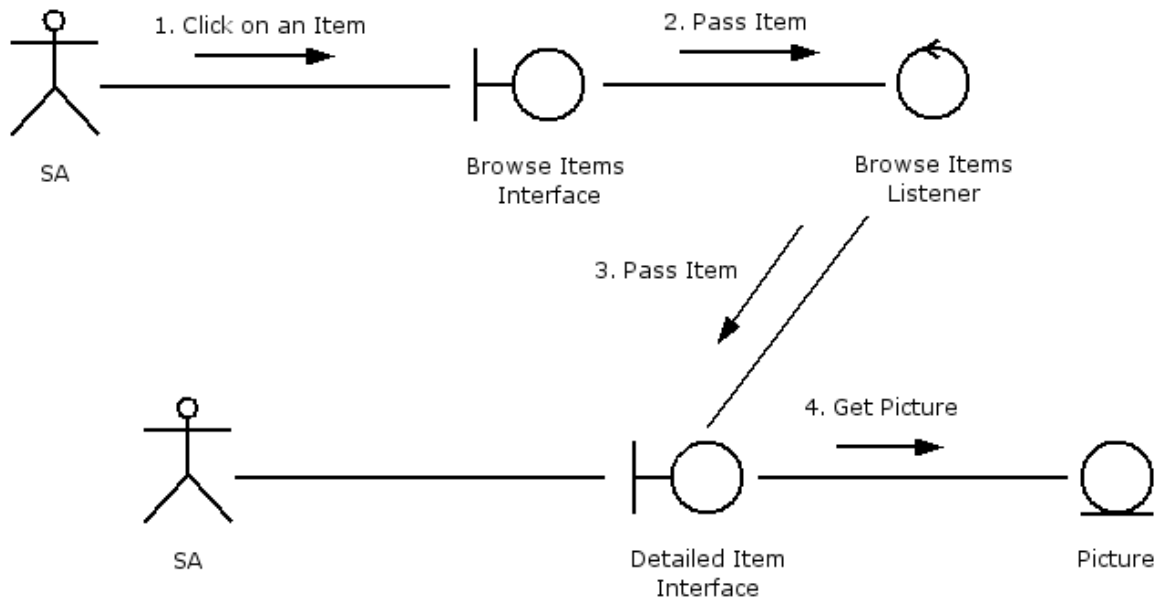


Figure 3.17 SA Browse collaboration diagram

1. The SA selects the Browse button on the Main Interface to show the Browse Item Interface.
2. The system shows a Browse Item Interface.

COMPLAINT

Normal Scenario

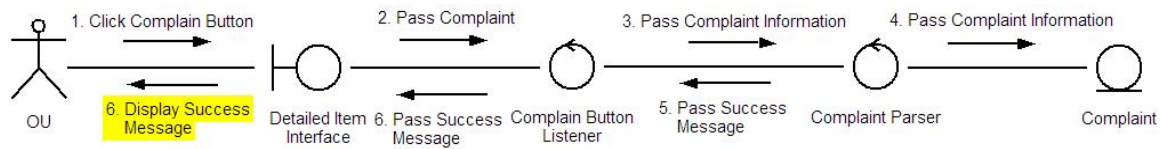


Figure 3.19 OU Complaint collaboration diagram

1. The OU has selected an item on the Browse Interface.
2. The OU selects the Complain button which shows the Submit New Complaint Interface.
3. The OU selects whether the complaint is against the owner of the auction or the auction item and enters their statement.
4. The OU selects the Submit Complaint button.
5. The system checks that there is something in the complaint text box.
6. The system creates an unapproved complaint.

Exceptional Scenario

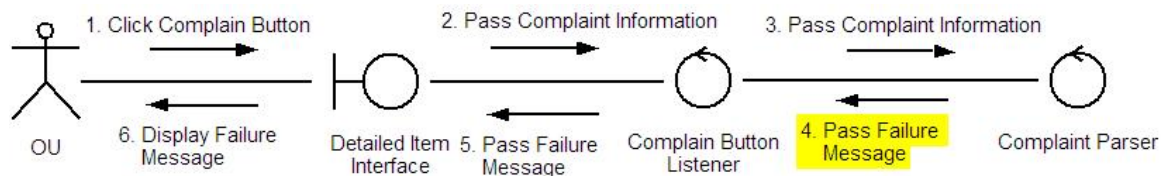


Figure 3.20 OU Complaint error collaboration diagram

1. The OU has selected an item on the Browse Interface.
2. The OU selects the Complain button which shows the Submit New Complaint Interface.
3. The OU selects whether the complaint is against the owner of the auction or the auction item and forgets to enter their statement.
4. The OU selects the Submit Complaint button.
5. The system checks and finds there is nothing in the complaint text box.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

6. The system then highlights the field label and prompts the user to correct the invalid field.
7. The OU fixes the error and selects the Submit Complaint button.
8. The system checks that there is something in the complaint text box.
9. The system creates an unapproved complaint.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

WITHDRAWAL

Normal Scenario

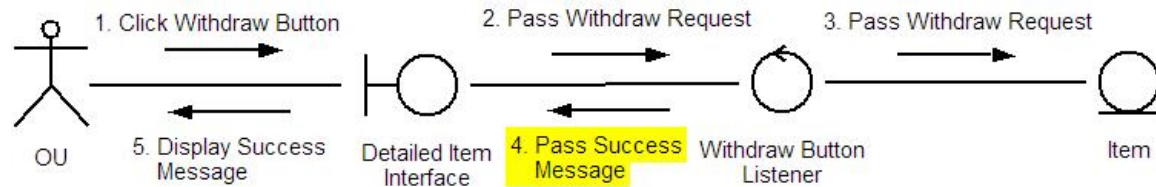


Figure 3.22 OU Withdraw an Item collaboration

1. The OU selects an item that they are selling in the Browse Interface.
2. The OU selects the Withdraw button.
3. The system updates the status of the selected item.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

SA Authorize

Normal Scenario

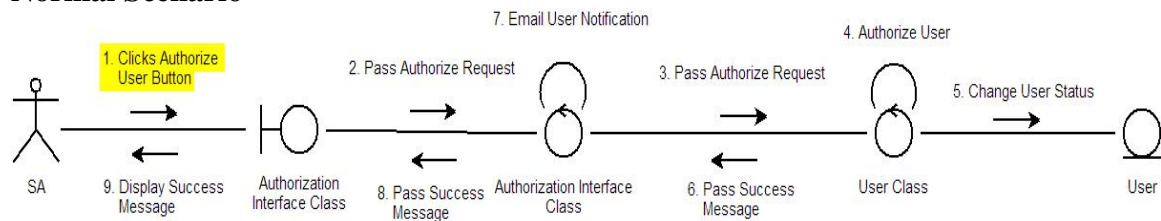


Figure 3.26 SA Authorize collaboration

1. The SA selects a user in the Authorization Interface.
2. The SA selects the Authorize button.
3. The system updates the status of the user and sends an email to the user with their username and password.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

SA Authorize Item

Normal Scenario

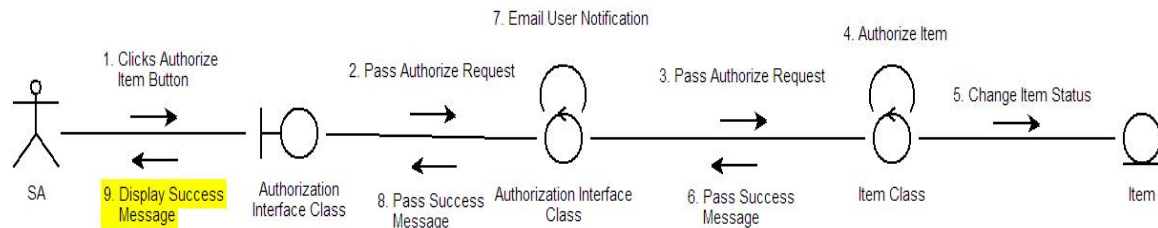


Figure 3.28 SA Authorize Item collaboration

1. The SA selects an item in the Authorization Interface.
2. The SA selects the Authorize button.
3. The system updates the status of the item and sends an email to the posting user notifying them that their auction has been approved and has started.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

SA Complaints

Normal Scenario

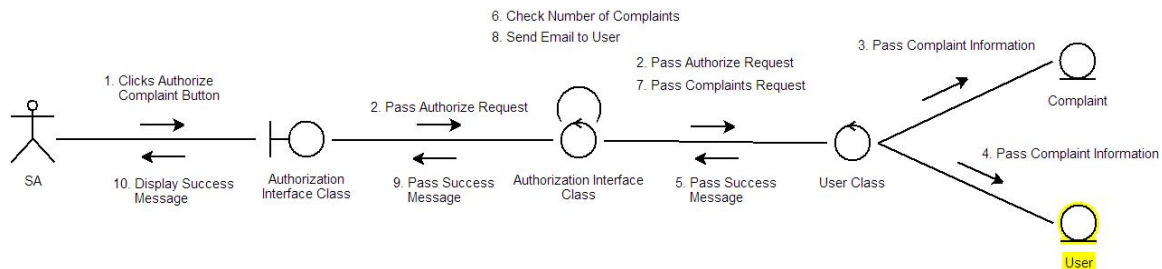


Figure 3.30 SA Complaints collaboration

1. The SA selects a complaint in the Authorization Interface.
2. The SA selects the Authorize button.
3. The system updates the status of the complaint and sends an email to the user notifying them that the complaint was processed.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

SA System Information

Normal Scenario

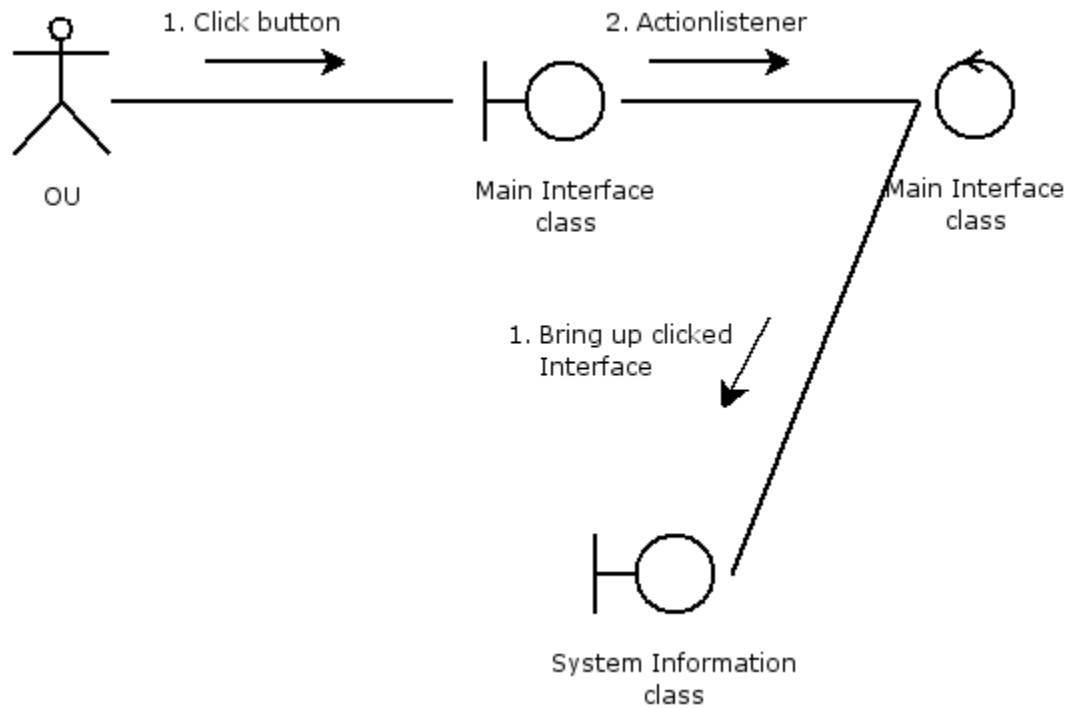


Figure 3.32 SA System Information collaboration

1. The SA selects the System Information button in the Main Interface which shows the System Information View.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

3. Detailed Design

Post Item Interface

Attributes

(Visual Elements)

- `private static final long serialVersionUID = 1L;`
- `private JPanel jContentPane = null;`
- `private JLabel jLabelTitle = null;`
- `private JLabel jLabelDescription = null;`
- `private JLabel jLabelStartingBid = null;`
- `private JLabel jLabelAuctionLength = null;`
- `private JButton jButtonSelectPicture = null;`
- `private JButton jButtonSubmit = null;`
- `private JTextField jTextFieldTitle = null;`
- `private JTextField jTextFieldDescription = null;`
- `private JTextField jTextFieldStartingBid = null;`
- `private JTextPane jTextPaneNote = null;`
- `private JComboBox jComboBox = null;`
- `private JComboBox jComboBox1 = null;`
- `private String auctionDays[] = { "1", "2", "3", "4", "5", "6", "7" };`
- `private selectPictureButton SelectPictureButton = new selectPictureButton();`
- `private JFileChooser selectPic = null;`
- `private String file = null;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private InputStream in = null;`
- `private String path = new String();`
- `private File selectedFile = null;`
- `private JLabel[] jLabel = { getJLabelTitle(),
getJLabelDescription(),
getJLabelStartingBid() };`

Functions

- `public PostItemInterface()`
`//This is the constructor for the Post Item Interface`
- `private class selectPictureButton implements ActionListener`
`// The action Listener for the select Picture Button`
- `public class pic extends FileFilter`
`//This is the filter that chooses which files are to be shown in
the select picture dialog`
- `public class jTextPostHandler implements ActionListener`
`//This handler checks all of the fields and makes sure the are
all filled in correctly`
- `private int getItemNumber()`
`//Get the item number for the newly posted item`
- `private JPanel getJContentPane()`
`//This function helps to create the GUI`
- `private JLabel getJLabelTitle()`
`//This Function creates the title label`
- `private JLabel getJLabelDescription()`
`//This function creates the description label`
- `private JLabel getJLabelStartingBid()`
`//This function creates the Starting bid label`
- `private JLabel getJLabelAuctionLength()`
`//This function creates the aucion length label`
- `private JButton getJButtonSelectPicture()`
`//This function creates the select picture button`
- `private JButton getJButtonSubmit()`
`//This function creates the submit button`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private JTextField getJTextFieldTitle()`
//This function creates the title field
- `private JTextField getJTextFieldDescription()`
//This function creates the the description field
- `private JTextField getJTextFieldStartingBid()`
//This function creates the starting bid field
- `private JTextPane getJTextPaneNote()`
//This function creates the panel note
- `private JComboBox getJComboBox1()`
//This function creates the auction dates drop down menu

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- ***Account Interface*** ---

Attributes

(Visual elements)

- private static final long serialVersionUID = 1L;
- private JPanel jPanel = null;
- private JLabel firstNameLabel = null;
- private JLabel lastNameLabel = null;
- private JLabel addressLabel = null;
- private JLabel cityLabel = null;
- private JLabel stateLabel = null;
- private JLabel zipCodeLabel = null;
- private JTextField firstNameTextField = null;
- private JTextField lastNameTextField = null;
- private JTextField addressTextField = null;
- private JTextField cityTextField = null;
- private JTextField stateTextField = null;
- private JTextField zipCodeTextField = null;
- private JButton updateInfoButton = null;
- private JButton changePasswordButton = null;
- private JPasswordField currentPasswordField = null;
- private JPasswordField newPasswordField = null;
- private JPasswordField confirmNewPasswordField = null;

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- private JLabel currentPasswordLabel = null;
- private JLabel newPasswordLabel = null;
- private JLabel confirmNewPasswordLabel = null;
- private JLabel creditCardLabel = null;
- private JTextField creditCardTextField = null;
- private JLabel phoneNumberLabel = null;
- private JTextField phoneNumberTextField = null;
- private JLabel userNameLabel = null;
- private JTextField userNameTextField = null;
- private String newPassword = new String();
- private UpdateButtonListener buttonPressedListener = new UpdateButtonListener();
- private PasswordButtonListener passwordButtonPressedListener = new PasswordButtonListener(); private JLabel emailLabel = null;
- private JTextField emailAddressTextField = null;
- private JLabel errorLabel = null;

Functions

- ClearFields
Clear all fields
- UpdateInfoButtonListener
If the fields have valid input ShowSuccess()
Else ShowError()
- ShowError
Highlight incorrect fields and update lblErrorNotice to prompt for re-entry
- ShowSuccess
Display "Information Updated Successfully"
- ChangePasswordButtonListener
Check if current password matches existing password for the user.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

If successful, check if new password entry matches confirm new password entry
Else ShowError()

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Browse Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private JPanel panelMain = null;`
- `private JScrollPane jScrollPaneMain = null;`
- `private JTable jTableData = null;`
- `private String[] colNames = null;`
- `private String[][] data = null;`
- `private String tableName;`
- `private String whereField;`
- `private String lookingFor;`
- `private MyTableModel tableModel = null;`

Functions

- `public BrowseInterface(String tableName_in, String whereField_in)`
`//This function grabs the information for the browse interface`
`and displays it on a table`
- `public JTable getJTable()`
`//This function gets the JTable`
- `private JPanel getPanelMain()`
`//This function gets the JPanel`
- `private JScrollPane getJScrollPaneMain()`
`//This function gets the scroll for the side of the panel`
- `private JTable getJTableData()`
`//This function gets the JTable`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private class ApproveButtonListener implements ActionListener`
`//This function responds by getting the item id when the approve`
`button is click.`
- `private void updateObject(int item_id) throws IOException,`
`Exception //This function updates the status on the user, item,`
`or compaint to approve.`
- `public void update(Observable arg0, Object arg1)`
`//This function updates the data on the table`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Detailed Item Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private JPanel pnlButton = null;`
- `private JButton btnPlaceBid = null;`
- `private JButton btnWithdrawItem = null;`
- `private JButton btnRepostItem = null;`
- `private JButton btnArchive = null;`
- `private JPanel pnlBid = null;`
- `private JLabel lblNumberOfBids = null;`
- `private JLabel lblNumberOfBidsValue = null;`
- `private JLabel lblCurrentPrice = null;`
- `private JLabel lblCurrentPriceValue = null;`
- `private JTextField txtBidAmount = null;`
- `private JLabel lblYourBid = null;`
- `private JPanel pnlItemDetails = null;`
- `private JLabel lblTitle = null;`
- `private JLabel lblDescription = null;`
- `private JLabel lblTitleValue = null;`
- `private JLabel lblDescriptionValue = null;`
- `private JButton btnImage = null;`
- `private item currentItem = null;`
- `private JLabel lblStartingPrice = null;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private JLabel lblStartingPriceValue = null;`
- `private JButton btnSubmitComplaint = null;`
- `private JLabel jLabelNotice = null;`
- `private JPanel jPanelContainer = null;`
- `private JLabel jLabelNotice2 = null;`

Functions

- `private class complaintButtonListener implements ActionListener`
`//This function controls the actions of the Complaint Button`
- `private class archiveButtonListener implements ActionListener`
`//This function Updates to current item`
- `private class repostButtonListener implements ActionListener`
`//This function reposts an item if it was removed before`
- `private class withdrawButtonListener implements ActionListener`
`//This function with draws a user from an item they bidded on.`
- `private class placeBidButtonListener implements ActionListener`
`//This function places a bid on an item.`
- `private void clearFields()`
`//This function clears all the text fields`
- `private void hideAllButtons()`
`//This function hides certain buttons`

```
private void updateManyObjects(String type, String matchField,
//This sets the notice label
private JLabel getJLabelNotice() {
}
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Authorization Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private JPanel jContentPane = null;`
- `private JButton jButApproveUser = null;`
- `private JButton jButApproveItem = null;`
- `private JButton jButApproveComplaint = null;`
- `private JTabbedPane jTabbedPane = null;`

Functions

- `public ApproveInterface(String approveType)`
`//This function adds a JPanel to the interface depending on the`
`approve type.`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Login Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private static systemInterface theSystemInterface = null;`
- `private static RegisterInterface theRegisterInterface = null;`
- `private JLabel jLabelUsername = null;`
- `private JLabel jLabelPassword = null;`
- `private JTextField jTextFieldUsername = null;`
- `private JPasswordField jPasswordFieldPassword = null;`
- `private JInternalFrame jInternalFrame = null;`
- `private JPanel jContentPane = null;`
- `private JButton jButtonLogin = null;`
- `private JButton jButtonRegister = null;`
- `private JLabel jLabelNote = null;`
- `private JLabel jLabelWelcome = null;`
- `private JLabel jLabelError = null;`
- `private boolean isValid = false;`
- `private ButtonPressedListener pressedButtonListener = new
ButtonPressedListener();`

Functions

- `public LoginInterface
//This function sets up the login interface GUI`
- `public void simulateSALogin(String username_in, String
password_in) //This function simulates a System Administrator
login`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `public void simulateOULogin(String username_in, String password_in) //This function simulates a Ordinary user login`
- `private class ButtonPressedListener implements ActionListener`
- `private void authenticateUser()
//This function checks if the user is in the system`
- `public void showError()
//this function displays any errors with user name not matching
or password not matching`
- `private JPanel getJContentPane()
//This function creates the look for login GUI`
- `private JButton getJButtonLogin()
//This function is for the login button`
- `private JButton getJButtonRegister()
//This function is for the register button`
- `private JLabel getJLabelNote()
//This function displays the note label`
- `private JLabel getJLabelWelcome()
//This function is for the welocme label`
- `private JLabel getJLabelError()
//This function is for the error label`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

TheMain Interface

Attributes

(Visual elements)

- `public static systemInterface theSystemInterface = null;`
- `private static LoginInterface loginInterface = null;`
- `public static Database global_db = null;`
- `public static user global_user = null;`

Functions

- `public static void createAndShowGUI() throws IOException, Exception //This function instantiates a new login interface and database`
- `public static void main(String[] args) throws IOException, Exception //This function instantiates createandshowgui function`
- `public static void showComplaintInterface(String user_id, String item_id) //This function instantiates complaint class function`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Register Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private JLabel jLabelFirstName = null;`
- `private JLabel jLabelLastName = null;`
- `private JLabel jLabelAddress = null;`
- `private JLabel jLabelCity = null;`
- `private JLabel jLabelState = null;`
- `private JLabel jLabelZip = null;`
- `private JLabel jLabelRequestedUsername = null;`
- `private JLabel jLabelCreditCard = null;`
- `private JLabel jLabelPhoneNumber = null;`
- `private JLabel jLabelEmail = null;`
- `private JPanel jPanelContentPane = null;`
- `private JTextField jTextFieldRequestedUsername = null;`
- `private JTextField jTextFieldFirstName = null;`
- `private JTextField jTextFieldLastName = null;`
- `private JTextField jTextFieldAddress = null;`
- `private JTextField jTextFieldCity = null;`
- `private JTextField jTextFieldState = null;`
- `private JTextField jTextFieldZipCode = null;`
- `private JTextField jTextFieldCreditCard = null;`
- `private JTextField jTextFieldPhoneNumber = null;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private JTextField jTextFieldEmail = null;`
- `private JButton jButtonSubmit = null;`
- `private JTextArea jTextAreaNote = null;`
- `private JTextField[] jFields = {
getJTextFieldRequestedUsername(),getJTextFieldFirstName(),
getJTextFieldLastName(),getJTextFieldAddress(),
getJTextFieldCity(), getJTextFieldState(),
getJTextFieldZipCode(), getJTextFieldPhoneNumber(),
getJTextFieldCreditCard(), getJTextFieldEmail() };`
- `private JLabel[] jLabels = { getJLabelRequestedUsername(),
getJLabelFirstName(), getJLabelLastName(), getJLabelAddress(),
getJLabelCity(), getJLabelState(), getJLabelZip(),
getJLabelPhoneNumber(), getJLabelCreditCard(), getJLabelEmail()
};`
- `private registerButtonHandler handler = new
registerButtonHandler();`

Functions

- `public RegisterInterface()
// This is the Button action listener for Submit. It checks all
of the`
- `private class registerButtonHandler implements ActionListener
// fields in the registration GUI`
- `private boolean notUniqueName(String userName)
// Checks if the user name is unique or not`
- `private JLabel getJLabelFirstName()
//This function creates first name label`
- `private JLabel getJLabelLastName()
//This function creates last name label`
- `private JLabel getJLabelAddress()
//This function creates address label`
- `private JLabel getJLabelCity()
//This function creates city label`
- `private JLabel getJLabelState()
//This function creates state label`
- `private JLabel getJLabelZip()
//This function creates zip code label`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- private JLabel getJLabelCreditCard()
//This function creates creditcard label
- private JLabel getJLabelPhoneNumber()
//This function creates phone number label
- private JLabel getJLabelEmail()
//This function creates email address label
- private JLabel getJLabelRequestedUsername()
//This function creates user name label
- private JPanel getJContentPane()
//This function creates GUI
- private JTextField getJTextFieldEmail()
//This function creates the email text field
- private JTextField getJTextFieldCreditCard()
//This function creates the creditcard text field
- private JTextField getJTextFieldPhoneNumber()
//This function creates the phone number text field
- private JTextField getJTextFieldRequestedUsername()
//This function creates the user name text field
- private JTextField getJTextFieldFirstName()
//This function creates the first name text field
- private JTextField getJTextFieldLastName()
//This function creates the last name text field
- private JTextField getJTextFieldAddress()
//This function creates the address text field
- private JTextField getJTextFieldCity ()
//This function creates the city text field
- private JTextField getJTextFieldState()
//This function creates the state text field
- private JTextField getJTextFieldZipCode()
//This function creates the zipcode text field
- private JButton getJButtonSubmit()
//This function creates the submit button
- private JTextArea getJTextAreaNote()
//This function creates the note text area

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Complaint Interface

Attributes

(Visual elements)

- `private static final long serialVersionUID = 1L;`
- `private JPanel jContentPane = null;`
- `private ButtonGroup radioButtonGroup = null;`
- `private JRadioButton userRadioButton = null;`
- `private JRadioButton itemRadioButton = null;`
- `private JLabel complaintAgainstLabel = null;`
- `private JLabel itemNumberLabel = null;`
- `private JLabel itemTitleLabel = null;`
- `private JLabel itemNumberDataLabel = null;`
- `private JLabel itemTitleDataLabel = null;`
- `private JTextArea complaintTextArea = null;`
- `private JLabel complaintLabel = null;`
- `private JButton submitButton = null;`
- `private String user_ID = null;`
- `private String item_ID = null;`
- `private String type = null;`
- `private String complaintText = null;`
- `private String title = null;`
- `private int userOrItem_id = 0;`
- `private int complainer_id = 0;`
- `private SubmitButtonListener submitButtonListener = new SubmitButtonListener();`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private radioButtonListener RadioButtonListener = new radioButtonListener();`
- `private ComplaintTextArea complaintTA = new ComplaintTextArea();`
- `private Element item_info;`

Functions

- `public ComplaintInterface(String user_ID, String item_ID)`
//This is the constructor class for complaint interface. This set the JContentpane,the global user ID, global item ID and the button for the class.
- `private String getItemTitle()`
// Gets the title of the auction, gets all the information on the item, and retrieves the title from the nodelist.
- `private class ComplaintTextArea implements FocusListener`
// This clears JTextArea if the error message is there
- `private class radioButtonListener implements ActionListener`
// This sets either the userRadioButton or ItemRadioButton, never both.
- `private class SubmitButtonListener implements ActionListener`
// This gathers all the information and passes it to complaint
- `private JPanel getJContentPane()`
//This function creates JPanel
- `private ButtonGroup getRadioButtonGroup()`
//Sets the radio button
- `private JRadioButton getUserRadioButton()`
//Set the user button
- `private JRadioButton getItemRadioButton()`
//Sets the item button
- `private JLabel getComplaintAgainstLabel()`
//Sets the Complaint text box title
- `private JLabel getItemNumberLabel()`
//Sets the item number title
- `private JLabel getItemTitleLabel()`
//Sets the Item title
- `private JLabel getItemNumberDataLabel`
//Sets the item number label

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- private JLabel getItemTitleDataLabel()
//Sets the item number title
- private JTextArea getComplaintTextArea()
//Sets the Complaint item text area
- private JLabel getComplaintLabel()
//Sets the complaint title
- private JButton getSubmitButton()
//Sets the submit button

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Transaction History Interface

Attributes

(Visual elements)

- `private BrowseInterface transactionHistoryInterface = null;`
- `private static final long serialVersionUID = 1L;`

Functions

- `public TransactionHistoryInterface()`
`//This function instantiates transactionHistoryInterface`
- `private BrowseInterface getTransactionHistoryInterface(){}
//This function instantiates BrowseInterface`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

User Class

Variables:

- `private int id;`
- `private int complaints;`
- `private boolean SA;`
- `private String status;`
- `private String userName;`
- `private String password;`
- `private String firstName;`
- `private String lastName;`
- `private String address;`
- `private String city;`
- `private String state;`
- `private String zipCode;`
- `private String creditCard;`
- `private String phoneNumber;`
- `private String emailAddress;`
- `private final int user_id = 0;`
- `private final int user_numComplaints = 3;`
- `private final int user_SA = 1;`
- `private final int user_status = 2;`
- `private final int user_password = 5;`
- `private final int user_userName = 4;`
- `private final int user_firstName = 7;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `private final int user_lastName = 8;`
- `private final int user_address = 9;`
- `private final int user_city = 10;`
- `private final int user_state = 11;`
- `private final int user_zipCode = 12;`
- `private final int user_creditCard = 6;`
- `private final int user_phoneNumber = 13;`
- `private final int user_emailAddress = 14;`

Functions:

- `public user(String newUsername, String newFirstName, String newLastName, String newAddress, String newCity, String newState, String newZipCode, String newCreditCard, String newPhoneNumber, String newEmailAddress) throws IOException, Exception`
`// Constructor for user class.`
`//This function takes in all of the user information and sets the global user information to the new given information.`
- `public user(String[][] data)`
`// Constructor is for loading purposes only`
- `public String[][] getFieldsAndDataForXML()`
`//This function sets up a two dimensional array. The first part of the array will contain the field names, the second part will contain the actual user information that corresponds to the field name. This double scripted array is then returned.`
- `public boolean canLogin`
`//This function checks if the user's status is approved; if it is they are allowed to log in. Else they are denied access.`
- `public String getInfo(String field)`
`// Getter for the attributes of the user class.`
- `public void setInfo(String field, String value)`
`// Setter for attributes of user class.`
- `public void incComplaint() throws IOException, Exception`
`//This function changes the user complaint status if a given complaint is approved against them. If they receive more than the 2 then they are blocked from the system.`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `public boolean authenticate(String username, String password)`
//This function checks that user exists. If they exist, they are able to log in, else, they are rejected.
- `public void register() throws IOException, Exception`
//This function registers the user in the database.
- `public void update() throws IOException, Exception`
//This function updates the given user information that was set.
- `public void changePassword() throws IOException, Exception`
//This function changes the user password to a newly requested password

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Item Class

Variables:

- `private int id;`
- `private String title;`
- `private String description;`
- `private long startPrice;`
- `private long currentPrice;`
- `private long startTime;`
- `private long endTime;`
- `private String pictureFileName;`
- `private int user_id;`
- `private String status;`
- `private final int item_id = 0;`
- `private final int item_title = 2;`
- `private final int item_startTime = 1;`
- `private final int item_currentPrice = 3;`
- `private final int item_endTime = 4;`
- `private final int item_user_id = 5;`
- `private final int item_startPrice = 6;`
- `private final int item_pictureFileName = 7;`
- `private final int item_description = 8;`
- `private final int item_status = 9;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Functions:

- `public item(String data[][])`
//Load constructor
- `public item(String newTitle, String newDescription, long newStartPrice, long newCurrentPrice, int newEndTime, String newPictureFileName, int newUser_id)`
// Constructor for the Item class.
- `public String[][] getFieldsAndDataForXML()`
//Returns a double scripted array. The first part of the array contains the field names. The second part of the array contains the corresponding information
- `public String getInfo(String field)`
// Getter of the attributes of the item class.
- `public void setInfo(String field, String value)`
// Setter of the attributes of the item class.
- `public void post() throws IOException, Exception`
//This function posts the information given in the info field
- `public void bid(Long currentBid) throws IOException, Exception`
//Allow a user to bid on an item. Once a bid is made, it is recorded as a transaction.
- `private long getHighestBid()`
// Gets the highest bid for this item, returns 0 if there are no bids for this item
- `public void withdraw() throws IOException, Exception`
//Changes the status of a users bid to withdrawn
- `public void update() throws IOException, Exception`
//To pass the updated item information to the database for storage.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Complaint Class

Variables:

- `private int id;`
- `private String status;`
- `private String type_of;`
- `private int userOrItem_id;`
- `private int complainer_id;`
- `private String complaintText;`
- `private int complaint_id = 0;`
- `private int complaint_type_of = 1;`
- `private int complaint_userOrItem_id = 2;`
- `private int complaint_complainer_id = 3;`
- `private int complaint_complaintText = 4;`
- `private int complaint_status = 5;`

Functions:

- `public complaint(String data[][])`
//Load Constructor
- `public complaint(String newType_of, int newUserOrItem_id, int newComplainer_id, String newComplaintText) throws IOException, Exception`
//Constructor of the Complaint Class.
- `public String[][] getFieldsAndDataForXML()`
//returns a double scripted array with the second part of the array containing the global variables. The first part of the array contains the attribute names.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `public String getInfo(String field)`
//Getter of the Complaint Class
- `public void setInfo(String field, String value)`
//This function takes in a string field name and a value and sets the given field name to that value
- `public boolean parse()`
//This function returns a boolean value depending on whether the complaint field has text or not.
- `public void submitComplaint() throws IOException, Exception`
//This function gets a new complaint number and submits the new complaint
- `public void update() throws IOException, Exception`
//To pass the updated item information to the database for storage.

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Database Class

Attributes:

- `private String[] fieldsInUser = { "id", "SA", "status", "complaints", "username", "password", "creditCard", "firstName", "lastName", "address", "city", "state", "zipCode", "phoneNumber", "emailAddress" };`
- `private String[] fieldsInComplaint = { "id", "type", "userOrItem_id", "complainer_id", "complaint_text", "status" };`
- `private String[] fieldsInItem = { "id", "startTime", "title", "currentPrice", "endTime", "user_id", "startPrice", "pictureFileName", "description", "status" };`
- `private String[] fieldsInTransaction = { "id", "user_id", "item_id", "amount", "timestamp", "type" };`
- `private String[] prettyFieldsInUser = { "Item #", "SA", "Status", "# Complaints", "Username", "Password", "Credit Card #", "First Name", "Last Name", "Address", "City", "State", "Zip Code", "Phone Number", "E-mail Address" };`
- `private String[] prettyFieldsInComplaint = { "Complaint #", "Type", "Complaint Against", "Complainer ID", "Complaint", "Status" };`
- `private String[] prettyFieldsInItem = { "Item #", "Start Time", "Title", "Current Price", "End Time", "Seller ID", "Start Price", "Picture", "Description", "Status" };`
- `private String[] prettyFieldsInTransaction = { "Transaction #", "User ID", "Item ID", "Amount", "Timestamp", "Type" };`
- `private static Document dom;`
- `private static final String dbFileName = "../xml/data.xml";`
- `private static Element rootElement = null;`
- `private static Node usersRootElement = null;`
- `private static Node itemsRootElement = null;`
- `private static Node complaintsRootElement = null;`
- `private static Node transactionsRootElement = null;`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Functions:

- `public Database() throws IOException, Exception`
//Constructor for the Database Class.
- `public String[][] select(String tableName, String whereField, String lookingFor, String placeHolder)`
//Allows for specified data selection from the xml file.
- `public Element select(String tableName, String whereField, String lookingFor)`
//Another Select function that returns a different type.
- `public void update(String[][] data)`
//This function updates the database information with the new information passed in.
- `public static Document createDocumentFromFile(String doc_name1)`
throws Exception, IOException
//To create a DOM from an existing xml file.
- `private static void printToFile()`
//This function flushes the DOM to file
- `private static String[] getItemInformation(Element itemElement)`
//This function grabs all the user information form the item information and returns it in a string array
- `private static String[] getUserInformation(Element userElement)`
//For a given user element get the values of the user information
- `private static String[] getComplaintInformation(Element complaintElement)`
//For a given user element get the values of the user information
- `private static String[] getTransactionInformation(Element transactionElement)`
//For a given user element get the values of the transaction information
- `private static String getText(Element ele, String tagName)`
// Changes the given element into a text based form
- `private static String getID(Element element)`
//Get the ID
- `public static int getMaxId(String tableName) throws IOException, Exception`
//This function returns the highest id in any given table
- `private static Element createTypeElement(String[][] data)`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

- `public void insert(String[][] data) throws IOException, Exception`
- `public String[] getArrayOfFieldsFor (String tableName)`
//This function returns the fields in a given table
- `public String[] getPrettyArrayOfFieldsFor (String tableName)`
//This function returns a different tyoe of field names for a given function
- `public void updateManyObjects(String type, String matchField, String matchValue, String[] updateFields, String[] updateValues) throws IOException, Exception`

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Transaction Class

Attributes:

```
private String type;

private int transaction_id = 0;

private int transaction_user_id = 1;

private int transaction_item_id = 2;

private int transaction_amount = 3;

private int transaction_timestamp = 4;

private int transaction_type = 5;
```

Functions:

```
//Load Constructor
//This function takes in a double string array with the second part of
the array being the item information.
public transaction(String data

//This function sets the newuser_id,newItem_id amount and type to the
given information
public transaction(int newUser_id, int newItem_id, long newAmount,
String newType) throws IOException, Exception

//This function returns a double-scripted array with the first part of
the array being the field types in string and the second part
containing the actual field type value
public String[][] getFieldsAndDataForXML()

//Gives the transaction an id plus adds all the correct fields to the
transaction table.
void storeTransaction() throws IOException, Exception

//sets the information for the given transaction fields
public void setInfo(String field, String value)

public void update() throws IOException, Exception
```


Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

ItemWatcher

Attributes:

```
private final int item_id = 0;

private final int item_endTime = 4;

Timer timer;
```

Functions:

```
//Constructor
public ItemWatcher()

//This function cancels the timer
public void cancel()

// Get the current time
public void run()
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

JTextFocusHandler

Attributes:

```
private JLabel[] jLabels;  
  
private JTextField[] jTextField;
```

Functions:

```
//Sets the global JLabel and JTextField  
public JTextFocusHandler(JLabel[] newJLabel, JTextField[]  
newJTextField)  
  
//This function erases errors  
public void focusGained(FocusEvent arg0)  
  
public void focusLost(FocusEvent arg0)
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

MyTableModel

Attributes:

```
private static final long serialVersionUID = 1L;

private String [][] data = null;

private String [] colNames = null;

private String [] colNamesNotPretty = null;

private int colCount;

private int rowCount;
```

Functions:

```
//This function creates a new table with the information that is
grabbed from the database.

public MyTableModel(String tableName_in, String whereField_in, String
lookingFor_in)

private void convertTimestamp(String fieldName){

//returns the column name
public String getColumnName(int col)

//returns the row count
public int getRowCount()

//This function returns the colmun count
public int getColumnCount()

//This function gets the data at a specified loction
public Object getValueAt(int arg0, int arg1)
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

SystemStatsInterface

Attributes:

```

private static final long serialVersionUID = 1L;

private JLabel jLabelNumOfOUs = null;

private JLabel jLabelNumOfOUsValue = null;

private JLabel jLabelItemsTotal = null;

private JLabel jLabelItemsTotalValue = null;

private JLabel jLabelTransactionsTotal = null;

private JLabel jLabelTransactionsTotalValue = null;

private JLabel jLabelComplaintsTotal = null;

private JLabel jLabelComplaintsTotalValue = null;

private JPanel jPanelUsers = null;

private JLabel jLabelUsers = null;

private JPanel jPanelTransactions = null;

private JPanel jPanelItems = null;

private JPanel jPanelComplaints = null;

private JLabel jLabelItems = null;

private JLabel jLabelTransactions = null;

private JLabel jLabelComplaints = null;

private JLabel jLabelItemsFinished = null;

private JLabel jLabelItemsFinishedValue = null;

private JLabel jLabelItemsActive = null;

private JLabel jLabelItemsActiveValue = null;

private JLabel jLabelItemsArchived = null;

private JLabel jLabelItemsArchivedValue = null;

private JLabel jLabelItemsWithdrawn = null;

```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

```

private JLabel jLabelItemsWithdrawnValue = null;

private JLabel jLabelUsersPending = null;

private JLabel jLabelUsersPendingValue = null;

private JLabel jLabelUsersApproved = null;

private JLabel jLabelUsersApprovedValue = null;

private JLabel jLabelUsersBlocked = null;

private JLabel jLabelUsersBlockedValue = null;

private JLabel jLabelComplaintsPending = null;

private JLabel jLabelComplaintsPendingValue = null;

private JLabel jLabelComplaintsApproved = null;

private JLabel jLabelComplaintsApprovedValue = null;

private JLabel jLabelItemsPending = null;

private JLabel jLabelItemsPendingValue = null;

```

Functions:

```

//This is the constructor function class
public SystemStatsInterface()

//This creates the number of OU label
private JLabel getJLabelNumOfOUs()

//This creates the number of OU value label
private JLabel getJLabelNumOfOUsValue()

//This creates the the JLabel
private JLabel getJLabel()

//This creates the number of items value label
private JLabel getJLabelNumOfItemsValue()

//This creates the number of transaction value label
private JLabel getJLabelNumOfTrans()

//This creates the number of transaction value label
private JLabel getJLabelNumOfTransValue()

//This creates the number of complaints label
private JLabel getJLabelNumOfComplaints()

//This creates the number of complaints value label

```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

```
private JLabel getJLabelNumOfComplaintsValue()

//This creates the JPanel for the users
private JPanel getJPanelUsers()

//This creates the label for the users
private JLabel getJLabelUsers()

//This creates the JPanel for the transactions
private JPanel getJPanelTransactions()

//This creates the JPanel for the items
private JPanel getJPanelItems()

//This creates the JPanel for the complaints
private JPanel getJPanelComplaints()

//This creates the items label
private JLabel getJLabelItems()

//This creates the Transaction label
private JLabel getJLabelTransactions()

//This creates the complaints label
private JLabel getJLabelComplaints()
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

TabbedPane

Attributes:

```
private static final long serialVersionUID = 1L;  
public BrowseItemsInterface browseItemsInterface = null;
```

Functions:

```
//Constructor for TabbedPane class.  
public TabbedPane()
```

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

4. System Screens

Please select an item and bid below

Item #	Start Time	Title	Current Price	End Time	Seller ID	Start Price	Picture	Description	Status
1	05/16/2006	A New Car	\$ 40	05/20/2006	admin	\$ 40	/item_images/...	It's shiny and p...	Approved
2	05/17/2006	Computer	\$ 101	05/18/2006	jgoldnight	\$ 100	/item_images/...	Antique machi...	Approved

Title:
Description:

Please enter a whole dollar amount greater than the current price

Bids:
Starting Price:
Current Price:
Your Bid:

Place Bid

Fig 4.1 TheMain Interface / Browse Interface

Submit Complaint Form

Complaint Against: ☐ User ☐ Item

Item #: 345

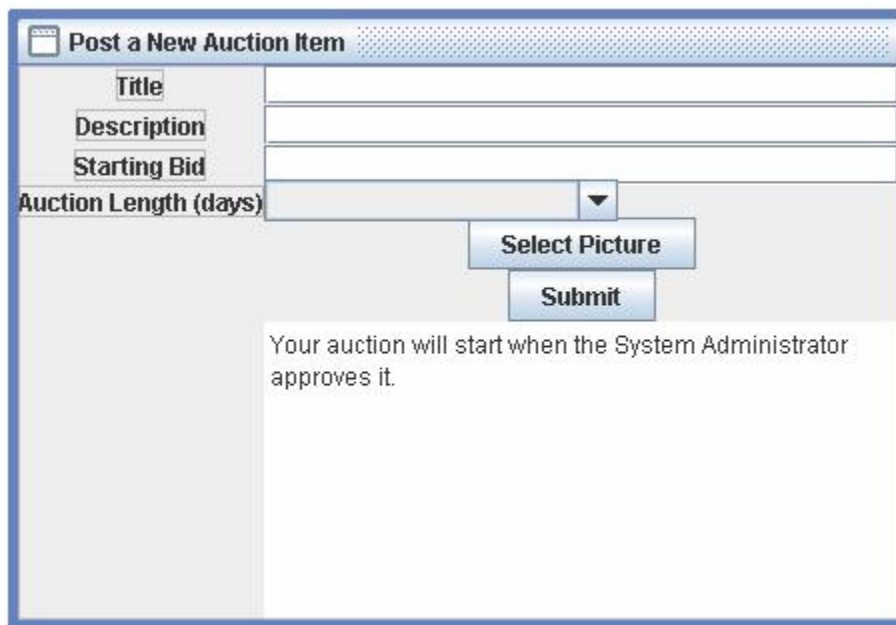
Item Title: Large boat

Complaint:

Submit

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Fig 4.2 Submit Complaint Interface



Post a New Auction Item

Title

Description

Starting Bid

Auction Length (days)

Your auction will start when the System Administrator approves it.

Fig 4.3 Post New Item Interface



Welcome to the CCNY Auction System
This is where the login error will show

Username


Password




Don't have an account? Click Register



Fig 4.4 Login Interface

CCNY Auction System

Main Interface

☒ Approve Items ☒ Approve Complaints  Statistics

 Post an Item  My Items  Transaction History ☒ Approve Users

 Browse Items  My Account


Username: admin
First Name: Admin
Last Name: System
Address: Please enter
City: Please enter
State: Please enter
ZIP Code: Please enter
Credit Card: Please enter
Phone Number: Please enter
E-mail Address: Please enter
Update Information




Current Password:
New Password:
Confirm New Password:
Change Password



Fig 4.5 My Account

CCNY Auction System

Main Interface

☒ Approve Items ☒ Approve Complaints  Statistics

 Post an Item  My Items  Transaction History ☒ Approve Users

 Browse Items  My Account

Item #	SA	Status	# Compla...	Username	Password	Credit Ca...	First Name	Last Name	Address	City	State	Zip Code	Phone Nu...	E-mail Ad...
3	false	Pending	0	blah	718-222-...	1234-12...	blah	blah	3333 ddd	ny	ny	10000	718-222-...	jj@jjg.com

Approve User

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Fig 4.6 Authorization

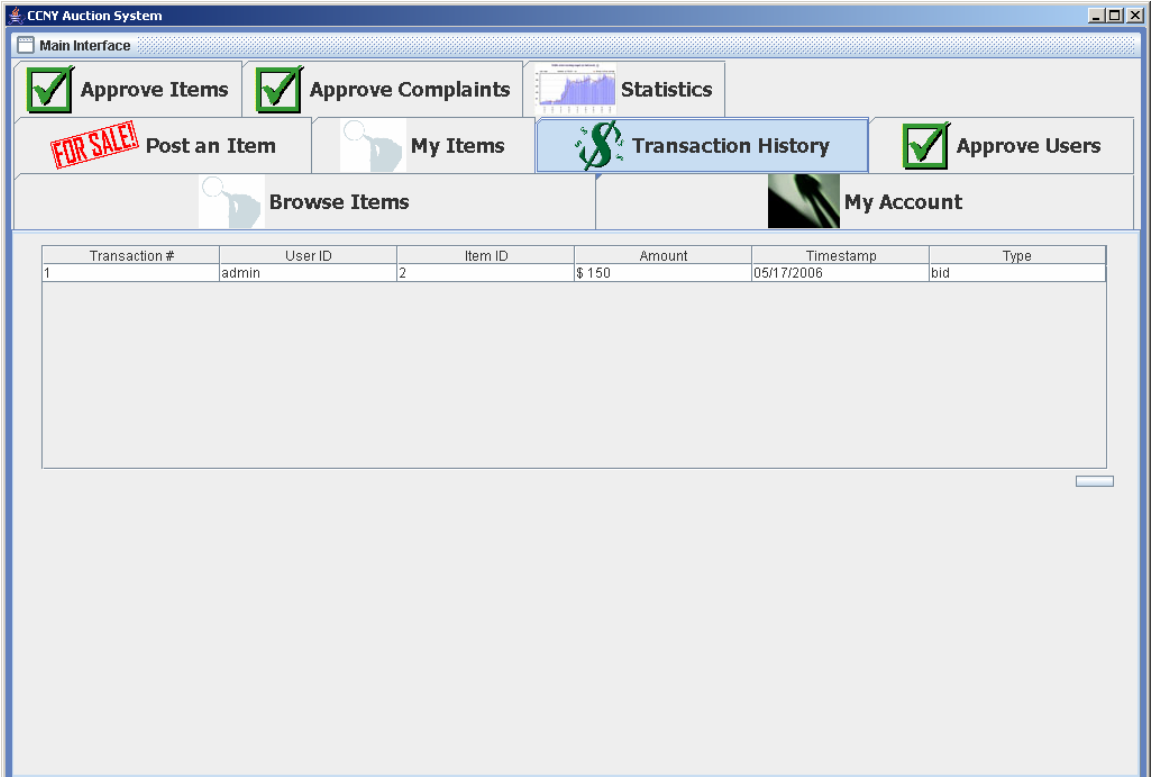


Fig 4.7 Transaction History

Electronic Bidding System	Version: 1.0
System Design Specification	Date: 5/17/06

Register

Requested Username

First Name

Last Name

Address

City

State

ZIP Code

PhoneNumber

CreditCard Number

Email Address

Submit

You will have access to the system
when the System Administrator
approves your account!

Fig 4.8 Registration