College of Saint Benedict & Saint John’s University

Computer Science Department

eSaint Aution System (eSAS)

Phase 3: Database Implementation

Team Name: Rahal Rampagers

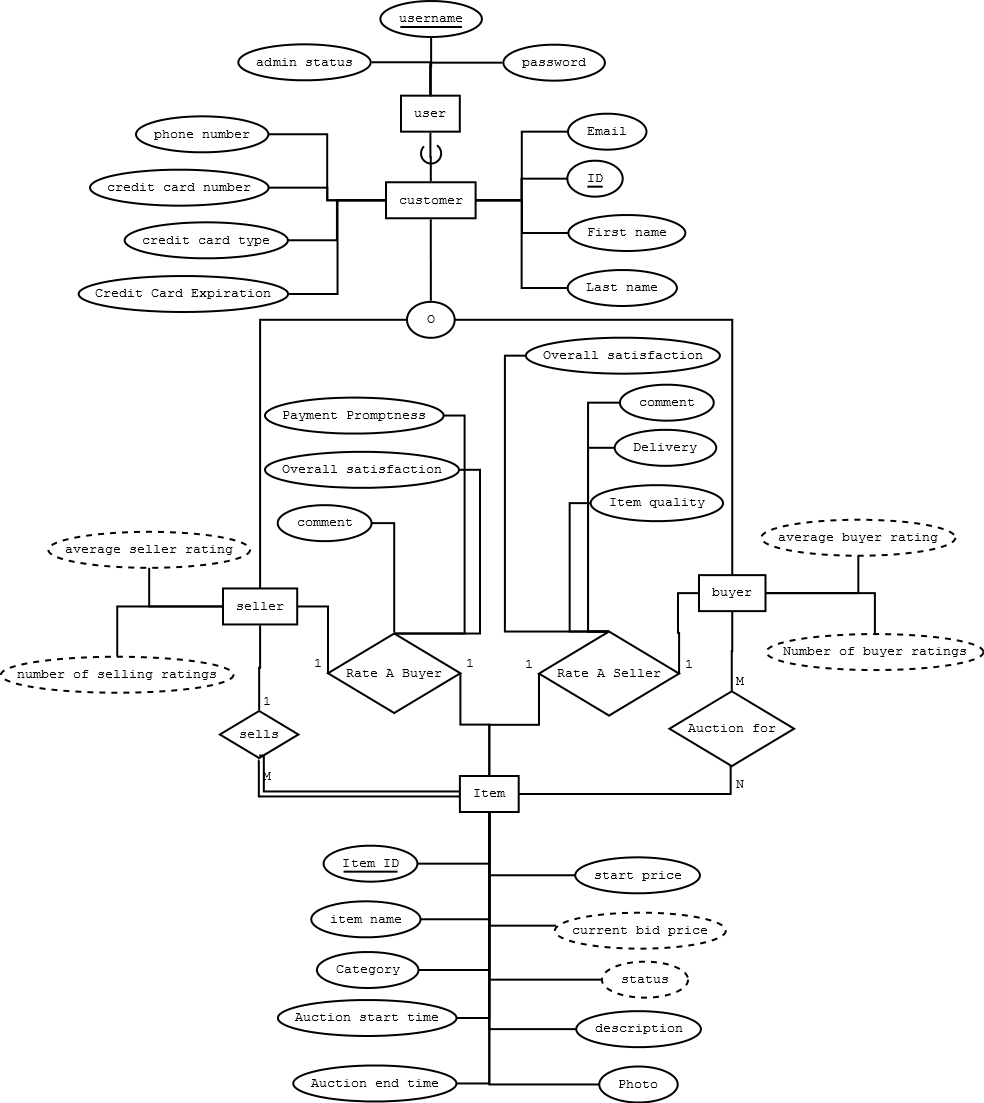
Justin Springer, Katie Kutzke, Phil Nowak, Tsuehue Xiong

11/10/2013

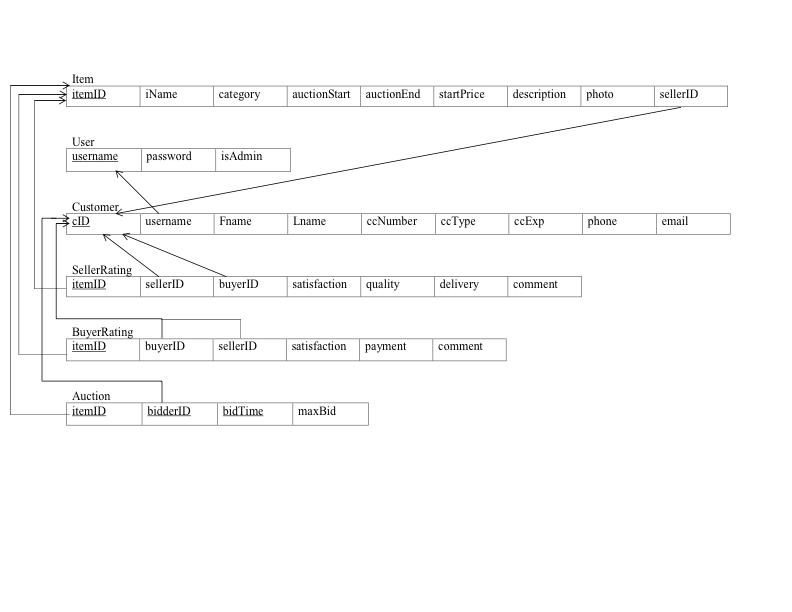
**Table of Contents**

EER Diagram……………………………………………………………………………………………………………………….……..3  
Relational Schema Diagram..……………………………………………………………………………………………………..4  
Create Table Statements……………………………..…………………………………………………………………………….5  
Populate Table Insert Statement………….………………………………………………………………………………….10  
Webpage Functionalities………………………………………………………………………………………………………….14  
SQL Code Component/Triggers….…………………………………………………………………………………………….32  
Problems/Issues………………………………………………………………………………………………………………………36  
Task Decomposition…………………………………………………………………………………………………………………37  
Meeting Minutes……………………………………………………………………………………………………………………..39  
 Meeting 9…………………………………………………………………………………………………………………….39  
 Meeting 10…………………………………………………………………………………………………………………..40  
 Meeting 11…………………………………………………………………………………………………………………..40  
 Meeting 12.………………………………………………………………………………………………………………….41  
 Meeting 13…………………………………………………………………………………………………………………..41  
 Meeting 14…………………………………………………………………………………………………………………..42  
 Meeting 15…………………………………………………………………………………………………………………..42  
 Meeting 16…………………………………………………………………………………………………………………..43  
 Meeting 17…………………………………………………………………………………………………………………..43  
 Meeting 18…………………………………………………………………………………………………………………..44

**EER Diagram**



**Relational Schema Diagram**



**Create Table Statements**

DROP SCHEMA IF EXISTS RahalRampagers;

-- Create the database

CREATE SCHEMA IF NOT EXISTS RahalRampagers;

use RahalRampagers;

-- Auction

CREATE TABLE `Auction` (

`itemID` int(11) NOT NULL,

`bidderID` int(11) DEFAULT NULL,

`maxBid` varchar(10) DEFAULT NULL,

`bidTime` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

PRIMARY KEY (`itemID`,`bidderID`,`bidTime`)

)ENGINE=InnoDB;

-- BuyerRating

CREATE TABLE `BuyerRating` (

`itemID` int(11) NOT NULL,

`sellerID` int(11) NOT NULL,

`buyerID` int(11) NOT NULL,

`itemName` varchar(10) NOT NULL DEFAULT '',

`satisfaction` int(11) NOT NULL DEFAULT '0',

`payment` int(11) NOT NULL DEFAULT '0',

`comment` varchar(50) DEFAULT NULL,

PRIMARY KEY (`itemID`)

)ENGINE=InnoDB;

-- SellerRating

CREATE TABLE `SellerRating` (

`itemID` int(11) NOT NULL,

`sellerID` int(11) DEFAULT NULL,

`buyerID` int(11) DEFAULT NULL,

`satisfaction` int(11) NOT NULL DEFAULT '0',

`quality` int(11) NOT NULL DEFAULT '0',

`delivery` int(11) NOT NULL DEFAULT '0',

`comment` varchar(50) DEFAULT NULL,

PRIMARY KEY (`itemID`)

)ENGINE=InnoDB;

-- Customer

CREATE TABLE `Customer` (

`cID` int(11) NOT NULL AUTO\_INCREMENT,

`username` varchar(15) NOT NULL,

`fName` varchar(10) DEFAULT NULL,

`lName` varchar(20) DEFAULT NULL,

`ccNumber` varchar(16) DEFAULT NULL,

`ccType` varchar(10) DEFAULT NULL,

`ccExp` varchar(10) DEFAULT NULL,

`phone` char(10) DEFAULT NULL,

`email` varchar(25) NOT NULL DEFAULT '',

`sellerRating` int(11) DEFAULT NULL,

`buyerRating` int(11) DEFAULT NULL,

PRIMARY KEY (`cID`)

)ENGINE=InnoDB;

-- User

CREATE TABLE `User` (

`username` varchar(15) NOT NULL UNIQUE,

`password` varchar(15) NOT NULL,

`isAdmin` tinyint(1) NOT NULL DEFAULT '0',

PRIMARY KEY (`username`)

)ENGINE=InnoDB;

-- Item

CREATE TABLE `Item` (

`itemID` int(11) NOT NULL AUTO\_INCREMENT,

`iName` varchar(50) NOT NULL DEFAULT '',

`category` varchar(15) NOT NULL DEFAULT 'none',

`auctionStart` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

`auctionEnd` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',

`startPrice` int(11) DEFAULT NULL,

`description` varchar(15) NOT NULL DEFAULT '',

`photo` tinyint(1) NOT NULL DEFAULT '0',

`sellerID` int(11) NOT NULL,

`currentBid` int(11),

-- current bid has to be something!!!!!!!!!!

`status` tinyint(2) default '0',

-- 0 is open auction, 1 is over but not sold, 2 is sold

PRIMARY KEY (`itemID`)

)ENGINE=InnoDB;

Alter Table Auction Add Constraint fk\_Auction\_bidderID foreign key (bidderID) references Customer(cID) on update cascade;

Alter Table Auction Add Constraint fk\_Auction\_itemID foreign key (itemID) references Item(itemID) on update cascade;

Alter Table BuyerRating Add Constraint fk\_BuyerRating\_sellerID foreign key (sellerID) references Customer(cID) on update cascade;

Alter Table BuyerRating Add Constraint fk\_BuyerRating\_buyerID foreign key (buyerID) references Customer(cID) on update cascade;

Alter Table BuyerRating Add Constraint fk\_BuyerRating\_itemID foreign key (itemID) references Item(itemID) on update cascade;

Alter Table SellerRating Add Constraint fk\_SellerRating\_sellerID foreign key (sellerID) references Customer(cID) on update cascade;

Alter Table SellerRating Add Constraint fk\_SellerRating\_buyerID foreign key (buyerID) references Customer(cID) on update cascade;

Alter Table Customer Add Constraint fk\_Customer\_username foreign key (username) references User(username) on update cascade;

Alter Table Item Add Constraint fk\_Item\_sellerID foreign key (sellerID) references Customer(cID) on update cascade;

**Populate Table Insert Statement**

*auction.dat:*

"2","1","8","2013-04-06 04:06:01"

"1","4","5","2013-04-06 04:06:01"

"3","3","5","2013-04-06 04:06:01"

"4","3","12","2013-04-06 04:06:01"

*buyer-rating.dat:*

"1","1","3","To Kill a Mocking Bird","1","1000000","I loved the book!"

"3","2","4","The Prestige","8","1","I enjoyed the movie."

*customer.dat:*

"","wsmith","william","smith","0123456789123456","AMEREXPRES","03/13","1112223333","wsmith@gmail.com",5,2

"","lsmith","liam","smith","0123456789123455","MASTERCARD","05/15","2223334444","lsmith@gmail.com",7,3

"","ismith","ill","smith","0123456789123455","CAPITALONE","06/14","2223334444","ismith@gmail.com",3,1

"","w1smith","will","smith","0123456789123454","VISA","08/13","3334445555","w1smith@gmail.com",9,8

*item.dat:*

"","To Kill a Mocking Bird","book","2013-04-02 04:06:01","2013-05-02 04:06:01","5","good","0","1","5","0"

"","To Kill a Mocking Bird","book","2013-11-02 04:06:01","2013-12-02 04:06:01","5","good","0","2","2","1"

"","The Prestige","Movie","2013-04-02 04:06:01","2013-05-02 04:06:01","5","OK","1","3","8","1"

"","Final Fantasy Tatics","game","2013-04-06 04:06:01","2013-05-08 04:06:01","7","very good","0","4","12","0"

"","The Lonesome Crowded West","CD","2013-04-06 04:06:01","2013-05-08 04:06:01","7","very good","0","1","12","2"

*seller-rating.dat:*

"1","3","1","10","1000000","10","I loved the book too!"

"3,"4","2","6","1","7","I did not enjoyed your business."

*user.dat:*

"wsmith","012345678912345","1"

"lsmith","applesauce","0"

"ismith","applepie","0"

"w1smith","applecrisp"

*load-auction.sql:*

LOAD DATA LOCAL INFILE "~/queries/auction.dat"

INTO TABLE Auction

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*load-buyerRating.sql:*

LOAD DATA LOCAL INFILE "~/queries/buyer-rating.dat"

INTO TABLE BuyerRating

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*load-customer.sql:*

LOAD DATA LOCAL INFILE "~/queries/customer.dat"

INTO TABLE Customer

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*load-item.sql:*

LOAD DATA LOCAL INFILE "~/queries/item.dat"

INTO TABLE Item

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*load-sellerRating.sql:*

LOAD DATA LOCAL INFILE "~/queries/seller-rating.dat"

INTO TABLE SellerRating

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*load-user.sql:*

LOAD DATA LOCAL INFILE "~/queries/user.dat"

INTO TABLE User

FIELDS ENCLOSED BY "\"" TERMINATED BY ","

;

*loadData.sql:*

SET foreign\_key\_checks=0;

source ~/queries/load-customer.sql;

source ~/queries/load-user.sql;

source ~/queries/load-sellerRating.sql;

source ~/queries/load-buyerRating.sql;

source ~/queries/load-item.sql;

source ~/queries/load-auction.sql;

SET foreign\_key\_checks=1;

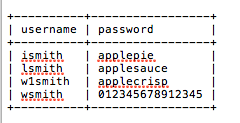
**Webpage Functionalities**

*Verbose Description and SQL Queries:*

*Login Page*

The User enters his/her username and password and hits the submit button. The username and password is compared to the other combinations in the database. If there is a match, the isAdmin boolean is received using a function with the given username and password as parameters, checked to determine what kind of User it is, and the User is redirected to the appropriate page. If there is not a match, the User is sent back to the login page with an appropriate error message.

-- login page  
select username, password   
from User;



select isAdmin  
from User  
where username = 'ismith' and password = 'applepie';



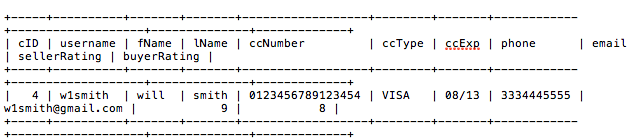
Customer Management Page

This is a hub page that includes buttons to view the userís own profile, the selling management, bidding management, and leave feedback pages.

View Profile

The user can view their profile information from this page. There are buttons on this page to update their profile and view their seller or buyer feedback that they have received.

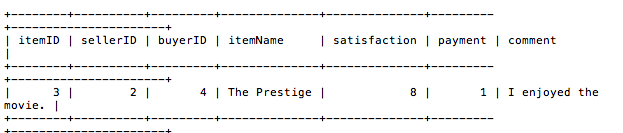
-- view your own profile  
select \*  
from Customer  
where cID = users cID  
-- Example cID = 4



My Buyer Feedback

This page displays buyer feedback information including the itemID, seller ID, buyer ID, item name, overall satisfaction, payment promptness, and a comment. These are results from sellers rating the current user as buyers of their items.

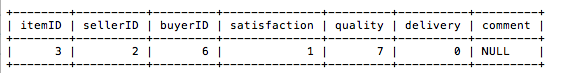
-- display buyer feedback  
select \*  
from BuyerRating  
--where buyerID = cID  
-- Example cID = 4



My Seller Feedback

This page displays all feedback that they have received from buyers of their item. This information includes the itemID, seller ID, buyer Id, item name, overall satisfaction, quality of item, delivery, and a comment.

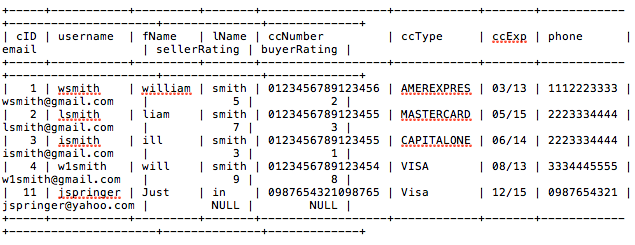
-- display seller feedback  
select \*  
from SellerRating  
-- Example cID = 2



Update Profile

The user can update any part of their profile except username, cID, buyer rating, and seller rating.  
This information will be displayed informs that can be changed and updated by hitting a button.

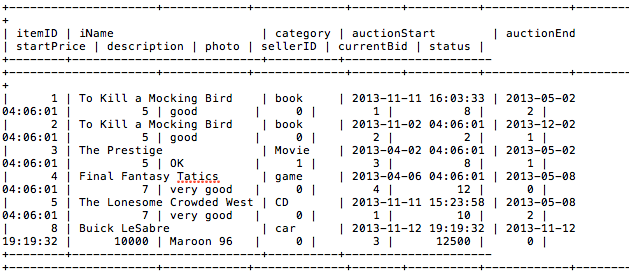
-- update profile  
update Customer   
set fName = newFName, lName = newLName, ccNumber = newccNumber,ccType = newccType, ccExp = newccExp,  
phone = newPhone, email = newEmail  
where cID = currentUser;  
-- Example cID = 11 and fName='Just', lName='in', ccNumber='0987654321098765', ccExp='12/15', phone='0987654321', email='jspringer@yahoo.com'



Selling Management

This hub page has buttons leading to add items to auction and view the list of items sold.  
Add Items  
The user inputs their item information and submits it to be put on auction.

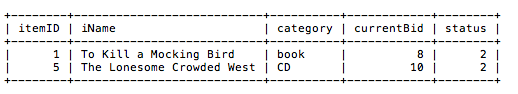
-- Add Items  
Insert into Item (iName, category, auctionStart, auctionEnd, startPrice, description, photo, sellerID, currentBid, status)  
-- Example values ('Buick LeSabre', 'car', now(), now(), '10000', 'Maroon 96', '0', '3', '12500', '0')



List Items on Auction

This page will list all the User's past and current auctions sorted by status.  
There will be links for each item that lead to the item info, bidder list, and rate buyer if someone bought that item.

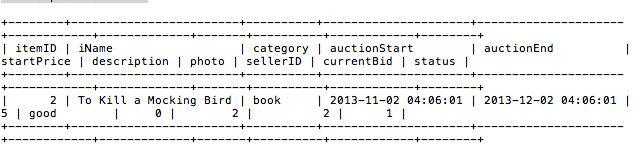
-- list items listed by seller (every item regardless of status)  
select itemID, iName, category, currentBid, status  
from Item   
-- where sellerID = current user  
order by status;  
-- Example sellerID = 1;



Item Info

This page displays the information about an item on auction. This includes itemID, item name, category, auction start and end times, the starting price, item description, photo, the seller ID number, current bid, and status of the item. There is also a button that lets the user bid on the item or see the bidding history of that item.

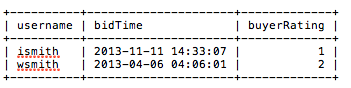
-- Item Info  
Select \*  
From Item  
-- where itemID = ?  
-- Example itemID = 2



Bidder List

This page will show just the usernames, buyer ratings, and bid times for bids on an item. It will not show the maximum bid, because this could be used unfairly. This list is shown while the item is currently on auction (status 0).

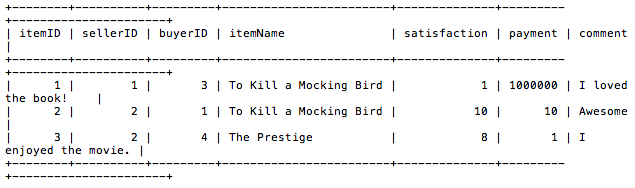
-- Bidder List (For Buyer)  
Select c.username, a.bidTime, c.buyerRating  
from Auction a, Customer c  
-- where itemID = current item being viewed and a.bidderID = c.cID  
order by bidTime Desc;  
-- Example itemID = 2



Rate Buyer

The customer that sold the item can rate the customer that bought the item.  
The user will input data for overall satisfaction and payment promptness into a form and hit the submit button.

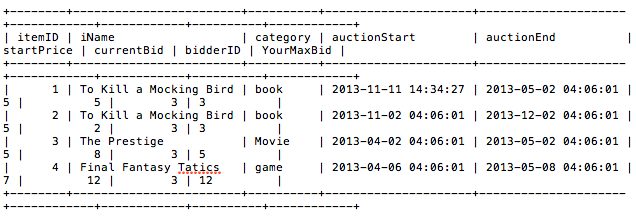
-- Rate Buyer  
Insert into BuyerRating Values ();  
-- Example Values ('2', '2', '1', 'To Kill a Mocking Bird', '10', '10', 'Awesome')



Bidding Management

This page has buttons leading to a list of items bid on and a search for items to buy.   
List of Items Bid On  
The page will display all items the User has bid on past and present sorted by itemID.

-- list items bid on  
select i.itemID, i.iName, i.category, i.auctionStart, i.auctionEnd, i.startPrice, i.currentBid, a.bidderID, a.maxBid as YourMaxBid  
from Item i, Auction a  
where a.bidderID = currentUser and i.itemID = a.itemID  
and bidTime = (select max(bidTime) from Auction where i.itemID = itemID and bidderID = currentUser);  
-- Example current user = 3



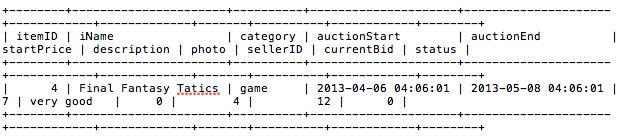
Search Items

The customer will input data into a search form and then hit the submit button.  
The page will forward the customer to a results page including links to items that fit their search criteria.

Results

The customer will see the results of their search query for items that are still available to bid on.  
The customer can then click on an item to view the auction profile for that item.

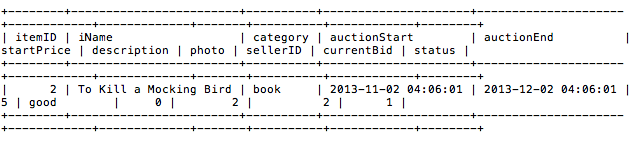
-- Search Item  
Select \*  
from Item  
-- where iName like %?% and category = ? and status = '0'  
order by auctionEnd;  
-- Example iName like '%final%' and category like '%%'



Item Profile

This page will display all the item information for the selected item. The customer can also view the seller's profile page, view the list of bids for the item, and bid on the item.

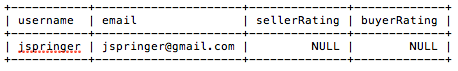
-- Item Profile  
Select \*  
From Item  
-- where itemID = ?  
-- Example itemID = 2



Profile Page

The user can view a person's profile that displays their username, email, seller rating, and buyer rating. The page will also have links to the seller's buyer and seller feedback.

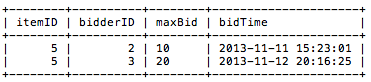
-- display users profiles  
select username, email, sellerRating, buyerRating  
from Customer  
-- where cID = specified user's ID  
-- Example cID = 8;



Bid On Item

The user will input a max bid and hit the submit button. This inserts the information into the Auction table and updates the current bid for the item.

-- Add New Bid  
Insert into Auction Values()  
-- Example Values ('5', '3', '20', now())



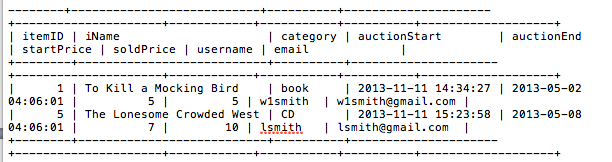
Leave Feedback

The User selects from two buttons that redirect them to pages where they can rate the sellers of items they have won and the buyers of items that they have sold.

List Of Items Sold

This page will display all items that have been sold (status=2) by the current User.  
The User can then select links to rate the buyer or view the item's info.

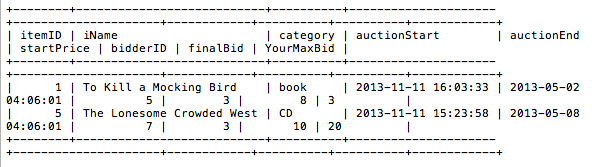
-- list items sold (auction status is ended and someone bought it)  
select i.itemID, i.iName, i.category, i.auctionStart, i.auctionEnd, i.startPrice, i.currentBid as soldPrice, c.username, c.email   
from Item i, Customer c, Auction a  
-- where i.sellerID = current user and status = '2' and a.itemID = i.itemID and c.cID = a.bidderID and a.maxBid >= i.currentBid  
-- Example current user = 1



List Of Items Bought

This page will display all items that have been bought by the current User.  
The User can then select links to rate the seller or view the item's info.

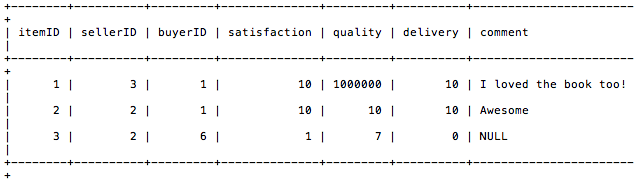
-- list items bought  
select i.itemID, i.iName, i.category, i.auctionStart, i.auctionEnd, i.startPrice, a.bidderID, i.currentBid as finalBid, a.maxBid as YourMaxBid  
from Item i, Auction a  
where a.bidderID = currentUser and i.itemID = a.itemID and i.status=2  
and bidTime = (select max(bidTime) from Auction where i.itemID = itemID and bidderID = currentUser);  
-- Example current user = 3



Rate Seller

The User that bought the item can rate the user they bought it from. The User will input data for each of the fields of a form and hit the submit button.

-- Rate Seller  
Insert into SellerRating Values ();  
-- Example Values ('2', '2', '1', '10', '10', '10', 'Awesome')



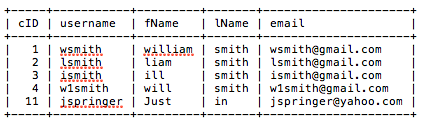
Admin Management

The admin has the option of two buttons: one shows a list of all the users and the other shows the reports page.

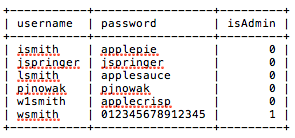
User List

Lists all the users by userID for the admin to see. This page also allows the admin to add a user.

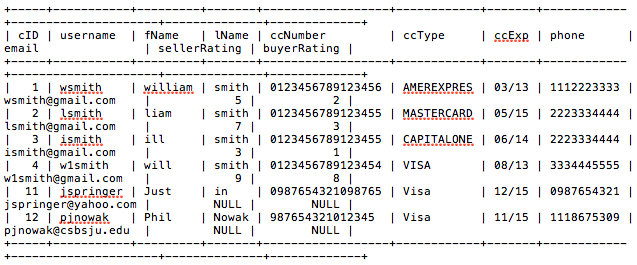
-- List users (for admin)  
SELECT cID,username,fName,lName,email FROM Customer;



-- Add user (for admin, can choose admin status. customers isAdmin = false)  
INSERT INTO User (username, password, isAdmin)  
VALUES -- Example Data ('pjnowak', 'pjnowak', 0)  
-- get values from form



INSERT INTO Customer (cID, username, fName, lName, ccNumber, ccType, ccExp, phone, email)  
VALUES -- Example Data ('', 'pjnowak', 'Phil', 'Nowak', '987654321012345', 'Visa', '11/15', '1118675309', 'pjnowak@csbsju.edu')  
-- get values from form



Reports

The admin has the option for two buttons: Overall commission and sales summary.

Overall commission lists all the users who have sold items and the amount of commissions for each

INSERT VIEWS HERE (SELECT \* FROM COMMISSIONREPORT)

Sales Summary

Lists all the items sold sorted by category and gives the commision and selling price for each.

INSERT VIEWS HERE(SELECT \* FROM SALESSUMMARYREPORT)

**SQL Code Component/Triggers**

BUYERRATING\_TRIGGER.sql: *drop trigger IF EXISTS BUYERRATING\_TRIGGER;  
delimiter &&  
CREATE TRIGGER BUYERRATING\_TRIGGER  
After INSERT on BuyerRating  
 FOR EACH ROW   
 Begin  
 declare satis int;  
 declare pay int;  
 declare buyerRat int;  
   
 set satis = (select Avg(satisfaction) From BuyerRating where   
 NEW.buyerID = BuyerRating.buyerID);  
   
 set pay = (select Avg(payment) From BuyerRating where   
 NEW.buyerID = BuyerRating.buyerID);  
 set buyerRat =(pay + satis) / 2;  
   
 UPDATE Customer  
 SET buyerRating = buyerRat where New.buyerID = Customer.cID;  
 End &&  
delimiter ;*

SELLERRATING\_TRIGGER.sql:

drop trigger IF EXISTS SELLERRATING\_TRIGGER;  
delimiter &&  
CREATE TRIGGER SELLERRATING\_TRIGGER  
After INSERT on SellerRating  
 FOR EACH ROW   
 Begin  
 declare satis int;  
 declare qual int;  
 declare del int;  
 declare sellerRat int;  
   
 set satis = (select Avg(satisfaction) From SellerRating where   
 NEW.sellerID = SellerRating.sellerID);  
   
 set qual = (select Avg(quality) From SellerRating where   
 NEW.sellerID = SellerRating.sellerID);  
   
 set del = (select Avg(delivery) From SellerRating where   
 NEW.sellerID = SellerRating.sellerID);  
 set sellerRat =(qual + satis+del) / 3;  
   
 UPDATE Customer  
 SET sellerRating = sellerRat where New.sellerID = Customer.cID;  
 End &&  
delimiter ;

*changePassword.sql:*

drop function if exists changePassword;  
delimiter &&  
Create function changePassword(name varchar(15),oldPassword varchar(15),newPassword varchar(15) ) Returns varchar(15)  
 Begin  
 declare curPassword varchar(15);  
 declare answer varchar(15);  
   
set curPassword = (select password from User where username = name);  
set answer = curPassword;  
if (oldPassword =curPassword) then  
 set answer = newPassword;  
 update User set password = newPassword where username = name;  
 End If;  
   
 return answer;  
   
 End   
 &&  
delimiter ;

*isAdmin.sql:*

drop function if exists isAdmin;  
delimiter &&  
Create function isAdmin(name varchar(15),pass varchar(15) ) Returns int  
  
 Begin  
  
 declare answer int;  
 set answer = (Select isAdmin from User where username = name and password = pass);  
  
 Return answer;  
 End   
 &&  
delimiter ;

**Problems/Issues**

No problems or issues.

**Task Decomposition**

Justin

-Worked on the ER Diagram

-Worked on the Webpages

-Worked on getting GitHub to work with Eclipse

-Gave up on Eclipse and got NetBeans to work with GitHub

-Worked on creating the Database Tables

-Find and make a list of functionalities for the Webpages

-Work on writing the SQL queries

-Worked on the database so that everything that is added in incremented and not set to a prefixed value

-Worked on SQL queries

-Worked on triggers

Katie

-Worked on the Relational Schema

-Worked on updating the Webpages

-Worked on creating the Database Tables

-Find and make a list of functionalities for the Webpages

-Work on writing the SQL queries

-Worked on the database so that everything that is added in incremented and not set to a prefixed value

-Worked on SQL queries

-Worked on triggers

Phil

-Worked on the Physical Database Design

-Worked on creating the Home Page

-Worked on creating the Database Tables

-Find and make a list of functionalities for the Webpages

-Worked on putting data into the database

-Worked on the database so that everything that is added in incremented and not set to a prefixed value

-Updated the insert data statements

-Worked on triggers

Tsuehue

-Worked on getting GitHub repository working

-Got everything relating to the project onto GitHub

-Worked on creating the Database Tables

-Find and make a list of functionalities for the Webpages

-Worked on how foreign key and primary key are connected

-Worked on putting data into the database

-Worked on the database so that everything that is added in incremented and not set to a prefixed value

- Worked on triggers

-Worked on the creating the report

**Meeting Minutes**

**Meeting 9 (10.4.2013) (1:30pm – 2:00pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**   
-review and go over the report  
  
 **HW:**  
 -Katie will work on the Relational Schema  
-Justin will work on the ER Diagram  
-Tsuehue will get GitHub working  
-Phil will work on the Physical database design –relational to MySQL mapping  
-Justin work on the webpages  
  
**Summary:**  
-We went over the last report and went through the requirements for Phase 3

**Meeting 10 (10.6.2013) (5:00pm – 6:30pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**   
-review and go over the report  
  
 **HW:**  
  
**Summary:**  
-Tsuehue worked on connecting all of our work onto GitHub  
-Phil worked on creating the Home Page  
-Justin worked on getting GitHub to work on Eclipse  
-Katie worked on updating the rest of the Webpages

**Meeting 11 (10.27.2013) (10:30pm – 12:00pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**  
  
**HW:**  
  
**Summary:**  
-Put NetBeans files onto GitHub  
-Went over Phase 3 requirements  
-Change meeting time (Sunday 10:30am, Wednesday 5:00pm, Monday 1:40pm)  
-Began creating the Database Tables

**Meeting 12 (10.28.2013) (1:40pm –3:00pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (absent)  
  
**Notes:**  
  
**HW:**  
  
**Summary:**  
-Worked on MySQL and created tables  
-Discuss about how we can make creating things in MySQL easier

**Meeting 13 (10.30.2013) (5:00pm –3:00pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**  
  
**HW:  
-**find and make a list of functionalities for the webpages (Justin)  
-find and make a list of functionalities for the webpages (Katie)  
-find and make a list of functionalities for the webpages (Philip)  
-find and make a list of functionalities for the webpages (Tsuehue)  
  
**Summary:**  
-Worked on MySQL and created tables with files  
-Tried to figure out why we get errorn 150

**Meeting 14 (11.3.2013) (10:30pm –12:30pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**   
  
**HW:**  
  
**Summary:**  
-Worked on MySQL and created tables with files  
-Talked about System Functionalities

**Meeting 15 (11.4.2013) (1:40pm –2:40pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**  
  
**HW:**-Justin will work on writing the SQL queries  
-Katie will work on writing the SQL queries  
-Phil will finish putting data into the database  
-Tsuehue will look up how foreign key and primary key are connected  
-Tsuehue will start working on the report  
  
**Summary:**  
-Talked about System Functionalities  
-Input Data into Database  
-Worked on SQL Query

**Meeting 16 (11.10.2013) (10:30pm –12:30pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**  
  
**HW:  
  
Summary:**  
-Updated the database so that everything that is added in incremented and not set to a prefixed value  
-Justin, Katie, Phil and Tsuehue worked on trigger  
-Justin and Katie worked on SQL queries  
-Phil updated the insert data statements

**Meeting 17 (11.11.2013) (1:40pm –2:40pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (here)  
  
**Notes:**  
  
**HW:**  
  
**Summary:**  
-Everyone worked on the triggers

**Meeting 18 (11.13.2013) (1:30pm –4:30pm)**  
**Attendance:**Justin Springer (here)  
Katherine Kutzke (here)  
Philip Nowak (here)  
Tsuehue Xiong (absent)  
  
**Notes:**   
  
**HW:**  
  
**Summary:**  
-We went over the report and made sure everything for Phase 3 was finished  
-Finish number 4 and 5 of Phase 3