UNIVERSITY OF PUERTO RICO
MAYAGÜEZ CAMPUS
ELECTRICAL AND COMPUTER ENGINEERING DEPARTMENT
ICOM 5016 – 036

# eMarket

Database Project Phase II
Professor: Manuel Rodriguez
November 11, 2013

Group Members
Samuel Matos
Victor J. Reventos
Eduardo M. Breijo

# DB Schema


Tables ..... 3

address_history ..... 3

admin_info ..... 3

bank_info ..... 3

bid_history ..... 3

billing_info ..... 3

cart_history ..... 4

cart_item_history ..... 4

category_info ..... 4

credit_card_info ..... 4

invoice_history ..... 4

invoice_item_history ..... 4

mailing_info ..... 5

notification_history ..... 5

payment_history ..... 5

products_drafts ..... 5

product_info ..... 5

product_quantity_record ..... 5

product_specification ..... 6

product_transaction_history ..... 6

question_answer_history ..... 6

question_history ..... 6

rating_history ..... 6

recently_viewed_items ..... 6

user_account_status ..... 7

user_info ..... 7

user_login_info ..... 7

Views ..... 8
active_users ..... 8
products ..... 8
report_params ..... 8
report_items ..... 8
report_day ..... 8
report_week ..... 8
report_month ..... 8

# Tables

### *address_history*

Address History Entity is a record of all addresses stored on the databases. It holds the address, country, city, geographical region and zip code of an address. It holds relationships with *mailing_info* and *billing_info* which communicate with *address_history* through foreign keys.

### *admin_info*

Admin Info Entity contains the records of all admins on the system. It holds the username, password, email, first name, middle name, last name, telephone, if an admin is a root admin and an account status flag that controls if the account is enabled or disabled ( since we don't want to delete any records from the database). It has no relationship with another table.

### *bank_info*

Bank Info Entity contains the record of the bank accounts of each user. It holds the user id, billing address id, bank name, bank account owner's name, account type (i.e. checking, savings, etc.), bank account number, bank account routing number and a status flag that determines if the account information is enabled or disabled. It holds relationships with *billing_info*, *user_id*, *payment_history* and *invoice_history* entities which communicate through foreign key references.

### *bid_history*

Bid History Entity contains the record of the bids occurring for each product on the system. It holds the user id, product id, bid amount, creation date and the close date (last bid that won the product). It holds relationships with *user_info* and *product_info* entities which communicate through foreign key references.

### *billing_info*

Billing Info Entity contains the record of all billing address for each user in the system. It holds the user id, address id, recipient name, telephone and a status flag that determines if the billing address is enabled or disabled. It holds relationships with *address_history*, *user_info*, *bank_info* and *credit_card_info* entities which communicate through foreign key references.

### *cart_history*

Cart History Entity contains the record of shopping carts each user has instantiated (i.e. add items and checkout) in the system. It holds the user id and the date it's closed (checkout). Each cart item is hold by another entity by the name of *cart_item_history*. Cart History Entity holds a relationship with *user_info* through foreign key references.

### *cart_item_history*

Cart Item History Entity contains the records of each item in an instance of shopping carts for each user. It holds the cart id, product id, quantity, creation date (date added) and closed date (checkout time). It contains relationship with *cart_history* and *product_info* through foreign key references.

### *category_info*

Category Info Entity contains the records of each category in the system. It holds category name, category parent id, creation date and status flag that determines if the category is enabled or disabled. It contains a relationship with itself (*category_info*) through a foreign key reference in order to assign a parent-child relationship.

### *credit_card_info*

Credit Card Entity contains the records of each credit card each user has in the system. It holds the user id, billing address id, card type, owner name, expiration date, card number, card csv and status flag that determines if the credit card is enabled or disabled (display in the user's payment options). It contains a relationship with *user_info*, *billing_info*, *payment_history* and *invoice_history* entities through foreign key references.

### *invoice_history*

Invoice History Entity contains the records of each purchase a user has done on the system. It holds the user id, bank id or card id paid with and the creation date. Each cart item is hold by another entity by the name of *invoice_item_history*. It contains a relationship with *user_info*, *credit_card_info*, *bank_info* and *invoice_item_history* entities through foreign key references.

### *invoice_item_history*

Invoice Item History entity contains each item of each purchase a user has done on the system. It holds the invoice id, product id, product quantity and the product sold price. It contains a relationship with *invoice_history* and *product_info* entities through foreign key references.

### *mailing_info*

Mailing Item Info entity contains the record of all mailing address for each a user in the system. It holds the user id, address id, recipient name, telephone, an *is_primary* flag that determines which mailing address is the one to be used for shipping and a status flag that determines if the mailing address is enabled or disabled. It holds relationships with *address_history* and *user_info* entities which communicate through foreign key references.

### *notification_history*

Notification History entity contains each notification a user has received by the system. It holds the user id, message, date (received) and an *is_read* flag that tells the system if the user has read the notification. It contains relationships with *user_info* through a foreign key reference.

### *payment_history*

Payment History entity contains each the transaction details for each payment and user does in the system. It contains the sender user id (sends money to *eMarket*), the recipient user id (*eMarket* sends money to), the amount of money of the transaction, the payment methods (Bank or Card), the card id or bank id the sender paid with, the transaction time and a *is_finished* flag to indicate that the system has finished the transaction. It contains a relationship with *user_info*, *bank_info* and *credit_card_info* entities which communicate through foreign key references.

### *products_drafts*

Product Drafts entity contains each draft of a product a user has saved in the system. It holds the user id, product specification id, creation date (date saved), update date (date updated) and closed date (date submitted to the product listing). It contains relationships with *user_info* and *product_specification* through foreign key references.

### *product_info*

Product Info entity contains the information of each product listed on the system. It holds the seller id, info spec id, creation date and depletion date (date when the product is not listed anymore). It contains relationships with *user_info*, *product_specification*, *recently_viewed_items*, *cart_item_history*, product_transaction_history, *invoice_item_history* and *bid_history* entities through foreign key references.

### *product_quantity_record*

Product Quantity Record entity holds the remaining quantity for each product on the system. It holds the remaining quantity. Each quantity record is related to a product specification id in *product_specification* entity though a foreign key reference.

### *product_specification*

Product Specification entity holds the details of each product listed on the system. It contains the category id, product name, non-bidding price (Buy It Now), starting bid price, bid end date (product end date), shipping price, quantity, description, condition (i.e., new, used), picture (system path to pic), brand, model, dimensions and a *is_draft* flag to specify the system if the product is listed or not. It contains a relationship with *category_info*, *product_quantity_record*, *product_drafts* and *product_info* entities through foreign key references.

### *product_transaction_history*

Product Transaction History entity holds each transaction in the system (user checkouts for each item). It contains product id, the quantity (quantity of the item) and the transaction date. It must be said that each transaction pertains to each item in the cart. It holds relationship with *product_info* entity through a foreign key reference.

### *question_answer_history*

Question Answer history entity holds the answers to the security questions a user fills at either registration time or profile update time. It holds the question id, user id, answer content and a status flag to determine if the answer is enabled or disabled in the system (depends on the security questions activated, max = 3 for now). It contains relationships with *user_info* and *question_history* entities through foreign key references.

### *question_history*

Question History entity holds the questions that will be used as a secured mechanism to retrieve a password for the user. It holds the question contents and it doesn't possess a foreign key reference to another table.

### *rating_history*

Rating History entity contains the history of ratings users have given each other in the system. It holds the user id of the rater, the user id of the person rated and the rating value given. It contains a relationship with *user_info* through a foreign key reference.

### *recently_viewed_items*

Recently Viewed Items entity contains a record of recently viewed items for each user. It holds the user id, product id, viewed date and a status flag that tells the system whether to display it on the system or not. It contains a relationship with *user_info* and *product_info* through foreign key references.

### *user_account_status*

User Account Status entity holds the status of each user account (disabled or enabled). It contains a relationship with the user id in *user_info* through a foreign key reference.

### *user_info*

User Info entity holds the record of each user registered in the system. It contains the first name, last name, middle name, telephone and creation time. It contains a relationship with many other entities which communicate through foreign key references.

### *user_login_info*

User Login Info entity contains login information for each user in the system. It holds the username, password and email for each user. It contains a relationship with user id in *user_info* through a foreign key reference.

# Views

### *active_users*

Active Users View is used to simplify the recovery of user account details for enabled accounts by creating a virtual entity from the *user_info*, *user_account_status* and *user_login_info* entities.

### *products*

Products view is used to simplify the recovery of many details of a product such as the current bid and other details relevant to the information of a product. This virtual entity is formed by using *bid_history*, *product_info* and *product_specification* entities and *active_users* virtual entity.

### *report_params*

Report Params view is used to denote the total operation cost and sales fee. The reports use the data in *report_params* in order to generate more virtual entities that help the creation of reports.

### *report_items*

Report Items view is used to recover the details necessary in order to produce the administrative report views. The logic to create this virtual entity is formed by using *invoice_history*, *invoice_item_history*, *product_info*, *product_specification* and *category_info* entities.

### *report_day*

Report Day view is used to recover the details necessary to produce administrative reports and calculate the revenue and sales per day. This virtual entity uses *report_items* and *report_params* views and *category_info* table in order to form the full report.

### *report_week*

Report Week view is used to recover the details necessary to produce administrative reports and calculate the revenue and sales per week. This virtual entity uses *report_items* and *report_params* views and *category_info* table in order to form the full report.

### *report_month*

Report Month view is used to recover the details necessary to produce administrative reports and calculate the revenue and sales per month. This virtual entity uses *report_items* and *report_params* views and *category_info* table in order to form the full report.