eMercado

An eBay clone web application

Anmol Reddy, Sai Koushik, Nayan Akarsh, Mavuri Manohar

180050059, 180050035, 180050074, 180050057

10.05.2021

CS-387: Database & Information Systems Lab

OVERVIEW OF REQUIREMENTS

We **intended** to design and implement an app which facilitates consumer-to-consumer sales like the platforms eBay, craigslist, OLX etc. do. (All users are equivalent and can both upload sales and buy others' sales. There are no "owner" or "admin" modes)

1) Users will be able to put items on sale
2) Others can search for them by name and get results in a ranked manner
3) They can also search for nearby products using location and tags for filtering.
4) Buyers can bid for them or buy them directly. There are auto-bidding modes when multiple users can set limits for their bids to be updated automatically.
5) Sellers can update the status of the delivery until the transaction is finally made when the buyer confirms the delivery.
6) Users can see the history of their current and old products, both bought and sold, their current status.
7) Communication between the parties during transactions
8) Give recommendations to users on products to buy
9) In employee and admin modes, they can edit user data to handle problems and perform data analysis.

WHAT WE DID *NOT* DO FROM ABOVE

We **did not implement:**

➢ *messaging* between users (as suggested by sir, it was orthogonal to the rest). We **instead added** a kind of message (which works like receipts) instead, so that a user has a receipt page with receipts generated for all the interactions in a transaction displayed.

➢ *recommenders* as that was too much out of our capacity (that requires other software and time constraints too).

➢ *range queries* using location, but **instead** used it for delivery modes and charges. We gave 4 modes, Light, Medium and Heavy Transport and Free Transport to find the delivery charges.

➢ *admin modes* as we needed a lot of running transactions to do data analysis and no need for an entire profile to do database error handling when we can **do it manually instead**.

➢ Product details and user details cannot be edited

USE CASE DESCRIPTIONS

1) Sign-up

- **Description:** New user creates account
- **Inputs:** name, username, password, re-enter password, location, email-id, phone number;
- **Pre-condition**: Person is not signed up and no one is logged in
- **Post-condition**: Person gets signed up, logged in into the new account and gets added into database (password hashed)
- **Main Success Path**: Creates a user account with the entered details, person table updated
- **Exceptions**: check for duplicate email, Verify password and re-entered password are same, if not refresh with other entered details

2) Logging in

- **Description**: log in to profile
- **Inputs**: email, password
- **Pre-condition**: Person is not logged in
- **Post-condition**: No change to database
- **Main Success Path**: Successfully logged in and we land on the user home page
- **Exceptions**: Incorrect password redirect, Request to re-enter the correct password

3) Adding Balance

- **Description:** User can add funds into his account
- **Inputs**: Credit Value
- **Pre-condition**: Account must exist and be logged in
- **Post-condition**: balance is updated in the database
- **Main Success Path**: Successfully credited the entered amount, and update balance
- **Exceptions**: Bad input — refresh and display an error message.

4) Withdraw money

- **Description**: User can withdraw funds from his account
- **Inputs**: Debit Value
- **Pre-condition**:Old balance is present in the database
- **Main Success Path**: Successfully debited the entered amount
- **Exceptions**: If excessive money requested — refresh with failed transaction
- **Post-condition**:Balance is updated in the database as an auction item

5) Adding a product for sale

- **Description**: Seller (User) can add a new item for sale
- **Inputs**: Product name, description, tags, base price (this is just price if it is auction), quantity and the type of transport needed for delivery (bike, truck, plane)
- **Pre-condition**: Seller (User) is logged in
- **Main Success Path**: Item is up for sale and available on searching
- **Exceptions**: redirect on invalid input
- **Post-condition**: Item is added to the database in the auction or direct items table

6) Search

- **Description**: Performs search over all items, direct and auction and gives the results in ranked order
- **Inputs**: Give a search term and any tag filters you want to search in
- **Pre-condition**: Seller (User) is logged in
- **Main Success Path**: Search results appear with links to open the product page to buy
- **Exceptions**: if no relevant results, search results are displayed as 0 results found
- **Post-condition**: No change to database

7) Select a product

- **Description**: Users can select a particular item from search results (this search results also appear when we find all the products used by someone, they come as search results)
- **Inputs**: None
- **Pre-condition**: search results are displayed
- **Main Success Path**: Selected item's details have to be displayed and options to open product
- **Exceptions**: None
- **Post-condition**: No change to database

8) Place a direct order

- **Description:** User can place direct order for the selected item
- **Inputs**: Click buy
- **Pre-condition**: Item is up for sale and selected by the buyer (user)
- **Main Success Path**: Order is placed successfully, on-hold balances updated, an order successful message is sent to the user.
- **Exceptions**: if price is not less than or equal to the difference of current balance and the on-hold balance then the order is not placed and an error message is displayed.
- **Post-condition**: Status of item is updated, on hold balance of the user is increased by the item price

9) Bid for an auction item

- **Description**: User can bid for an item
- **Inputs**: bid value, bid mode (auto option), limit (optional - needed for auto case);
- **Pre-condition**:Item is up for sale and selected by the buyer (user)
- **Main Success Path**: Bid is placed successfully, best bid and associated on-hold balances updated, a bid successful message is sent to the user.
- **Exceptions**: if bid value and limit are not less than or equal to the difference of current balance and the on-hold balance then bid is not placed and an error message is displayed.
- **Post-condition**: Bid is added to the database and on-hold balance is updated based on auto-bids made, best bid may be updated in the item attributes

10) Remove a sale

- **Description**: Seller can remove the item from website
- **Inputs**: button click
- **Pre-condition**: product is present in the database, belongs to current user
- **Main Success Path**: Product removed from availability (no longer appears in search results)
- **Exceptions**: none
- **Post-condition**: product status changed to “closed”

11) View bids

- **Description**: Displays all the bids of an user
- **Inputs**: View bids button to be clicked.
- **Pre-condition**: None
- **Main Success Path**: Displays all the bids of the user.
- **Exceptions**: None
- **Post-condition**: No change to database

12) View sales

- **Description**: Displays all the sales of an user both closed and sold
- **Inputs**: View sales button to be clicked.
- **Pre-condition**: None
- **Main Success Path**: Displays all the sales of the user.
- **Exceptions**: None
- **Post-condition**: No change to database

13) View purchases and orders

- **Description**: Displays all the purchased products and current orders by an user
- **Inputs**: View my purchase button to be clicked.
- **Pre-condition**: None
- **Main Success Path**: Display all the direct orders and purchased products of the user.
- **Exceptions**: None
- **Post-condition**: No change to database

14) Update a bid

- **Description**: Updates mode of bid or value of the bid
- **Inputs**: New values of the same fields used for initial bids;
- **Pre-condition**: Old details of the bid are present in the database
- **Main Success Path**: Bid details have to be updated
- **Exceptions**: Bid value cannot be decreased — refresh and an error message gets displayed
- **Post-condition**:New details of the bid is updated in the database

15) Closed bidding and make transaction with best bidder

- **Description**: Seller (User) can end the bidding of the product
- **Inputs**: button click; At least 1 bid must have been made so far
- **Main Success Path**: Bidding has to be ended and accepted, buyer and seller get receipts on their sale and bids
- **Exceptions**: If no bid is present then the item remains unsold with error displayed

- **Post-condition**: Status of item (sold) and bids are updated accordingly in their tables

16) Update delivery status

- **Description**: Seller can update the item's status, buyer is able to confirm delivery
- **Inputs**: set to new value of status; It can be either Auctioned–→ Shipping–→ Shipped and go–→ out-for-delivery
- **Pre-condition**: Bidding closed already and not out-for-delivery or higher
- **Main Success Path**: Status of the item has to be updated and if out-for-delivery, the seller can make no more changes but the buyer is now able to confirm the delivery
- **Exceptions**: If already out-for-delivery or unsold, cannot be updated
- **Post-condition**: New product status is updated in the database

17) Buyer confirms delivery

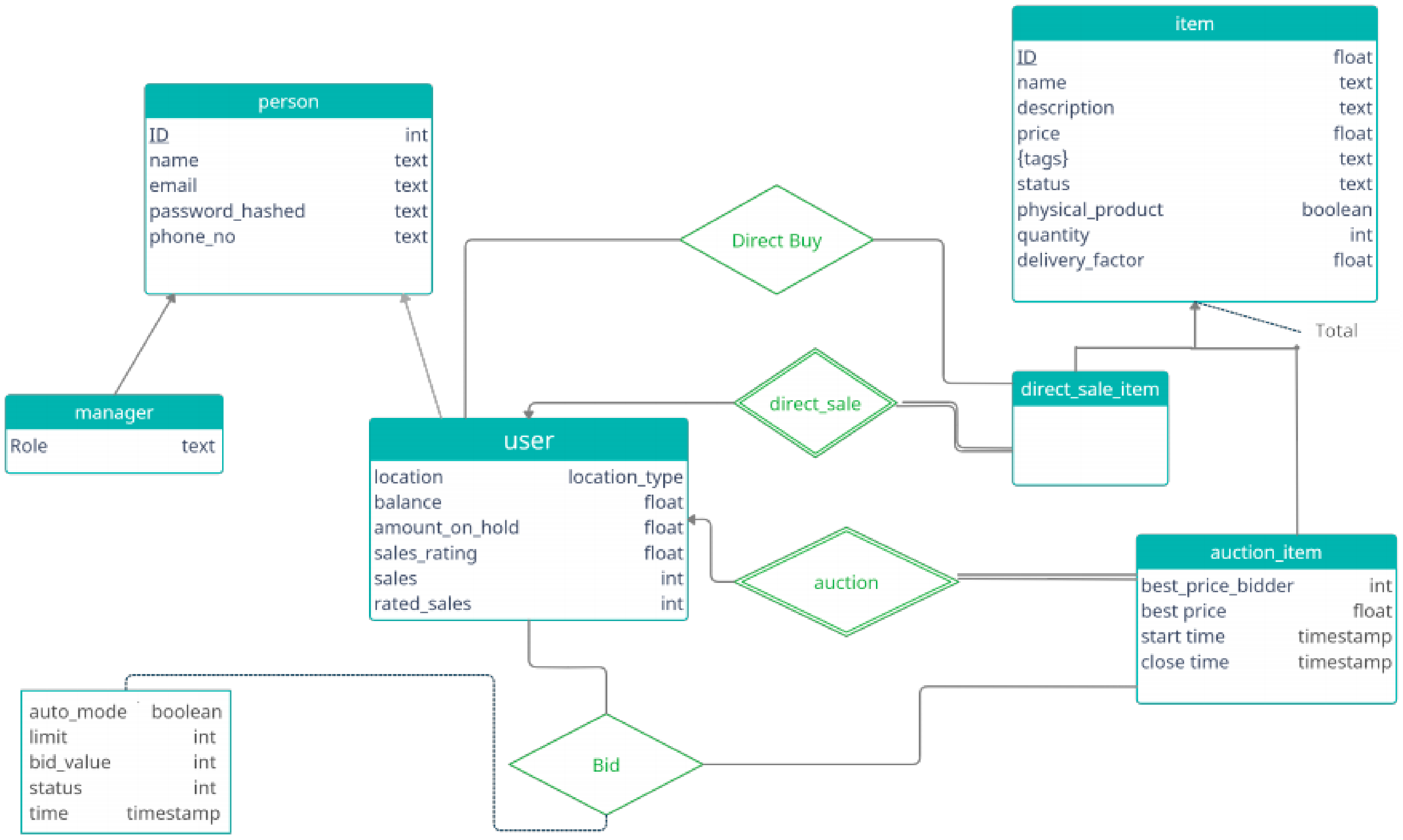
- **Description**: Buyer can update the item's status to Delivered and make the transaction
- **Inputs**: user (buyer) button click in product page
- **Pre-condition**: Item was in out-for-delivery status
- **Main Success Path**: Status of the item has to be updated to delivered, transactions completed, receipts received, balances and on-hol balances updated
- **Exceptions**: none
- **Post-condition**: Buyer and Seller's balance has to be updated, and Buyers on-hold balance has to be updated, status update, message receipts generated for both

18) View messages

- **Description**: Displays all the messages of the user
- **Inputs**: View message button to be clicked.
- **Pre-condition**: None
- **Main Success Path**: Displays all the messages of the user in sorted order using timestamp of message.
- **Exceptions**: None
- **Post-condition**: No change to database

DESIGN

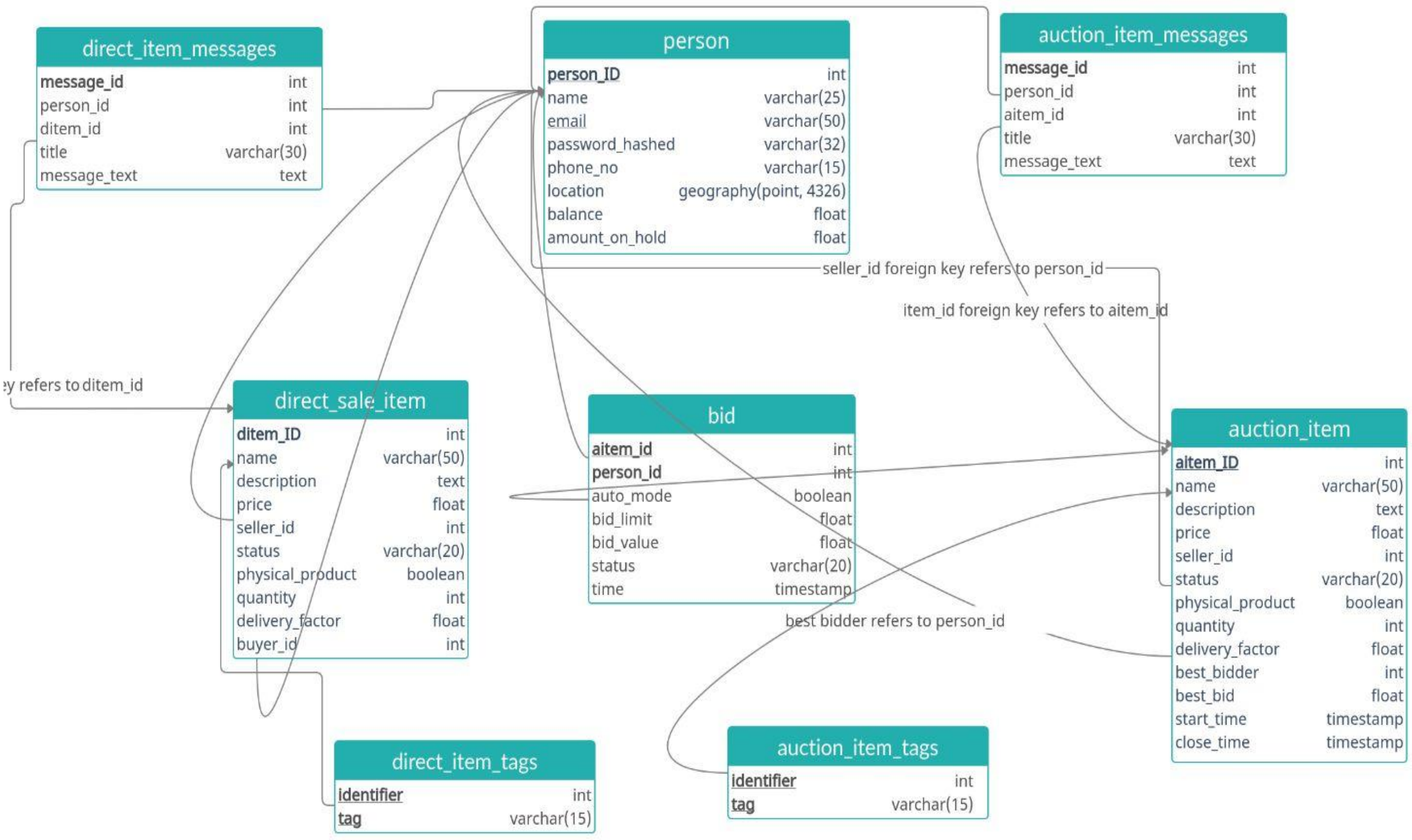
This was our initial ER diagram, to this later we added *message* entities and made all *person*s into *user*s. We removed some attributes like *physical_product* and we normalized into the schema diagram given in the next page. We split *item* into two tables; dealing with directly sold and auctioned items separately after normalization. We added *tags* entity too, to help with classification in search filters.



Our initial schema in deliverable 3 did not have these indices relevant to our queries. We now have added indices on account of our queries:

1. Indexing *tag* in our *tags* tables - because we use filtered-search queries which check by *tag* value.
2. Indexing on *person_id* in *messages* because we fetch messages of a user by his *person_id*
3. In the *item* tables we indexed *seller_id* because we fetch all the items relevant to a particular seller in several queries (like in balance updates, fetching all his bought and sold products etc).

We removed some now-useless attributes (like physical_product) and added 2 tables for message receipt handling. We did not have any need of denormalization as our joins always involved only 2 tables at most.



FINAL DESIGN

The associated DDL file is in the repository as `eMercadoDDL.sql`

```
DROP table if exists direct_item_messages;
DROP table if exists auction_item_messages;
DROP TABLE if exists bid;
DROP TABLE if exists direct_sale_item_tags;
DROP TABLE if exists auction_item_tags;
DROP TABLE if exists direct_sale_item;
DROP TABLE if exists auction_item;
DROP TABLE if exists person;

CREATE TABLE person (
      person_id serial,
      name varchar(25),
      email varchar(50) not null unique,
      password_hashed varchar(64),
      phone_no varchar(15),
      location geography(point,4326),
      balance float,
      amount_on_hold float check(amount_on_hold<=balance),
      primary key(person_id)
);



CREATE TABLE auction_item (
      aitem_id serial,
      name varchar(150),
      description text,
      price float,
      seller_id int not null,
      status varchar(20) check(status='open' or status='closed' or status='auctioned' or
            status='shipping' or status='shipped' or status='out-for-delivery' or
status='delivered'),
      physical_product boolean,
      quantity int default 1 check(quantity>0),
      delivery_factor float check(delivery_factor>=0),
      best_bidder int,
      best_bid float check(best_bid is null or best_bid>=price),
      start_time timestamp,
      close_time timestamp check(close_time>start_time),
      primary key(aitem_id),
      foreign key(best_bidder) references person on delete set null,
      foreign key(seller_id) references person on delete cascade
);

CREATE TABLE direct_sale_item (
      ditem_id serial,
      name varchar(150),
      description text,
      price float,
      seller_id int not null,
      status varchar(25) check(status='open' or status='closed' or status='sold' or
            status='shipping' or status='shipped' or status='out-for-delivery' or
status='delivered'),
      physical_product boolean,
      quantity int default 1 check(quantity>0),
      delivery_factor float check(delivery_factor>=0),
      buyer_id int,
      primary key(ditem_id),
      foreign key(seller_id) references person on delete cascade,
      foreign key(buyer_id) references person on delete set null
);

CREATE TABLE auction_item_tags (
      identifier int,
      tag varchar(100),
      primary key(identifier,tag),
      foreign key(identifier) references auction_item on delete cascade
);

CREATE TABLE direct_sale_item_tags (
      identifier int,
      tag varchar(100),
      primary key(identifier,tag),
      foreign key(identifier) references direct_sale_item on delete cascade
);

-- need to ensure that on-hold balances are updated when products kept on sale are deleted
CREATE TABLE bid (
      aitem_id int,
      person_id int,
      auto_mode boolean,
      bid_limit float,
      bid_value float,
      status varchar(20) check(status='rejected, removed' or status='running' or
status='accepted'),
      time timestamp,
      primary key(aitem_id,person_id),
      foreign key(aitem_id) references auction_item on delete cascade,
      foreign key(person_id) references person on delete cascade
);

CREATE TABLE direct_item_messages(
      message_id  serial,
      person_id   int,
      ditem_id    int,
      title       varchar(30),
      message_text text,
      message_time timestamp,
      primary key(message_id),
      foreign key(ditem_id) references direct_sale_item on delete cascade,
      foreign key(person_id) references person on delete cascade
);

CREATE TABLE auction_item_messages(
      message_id  serial,
      person_id   int,
      aitem_id    int,
      title       varchar(30),
      message_text text,
      message_time timestamp,
      primary key(message_id),
      foreign key(aitem_id) references auction_item on delete cascade,
      foreign key(person_id) references person on delete cascade
);

CREATE index receipt_user_1 ON direct_item_messages(person_id);
CREATE index receipt_user_2 ON auction_item_messages(person_id);
CREATE INDEX auslr_idx ON auction_item(seller_id);
CREATE INDEX dsslr_idx ON direct_sale_item(seller_id);
CREATE INDEX dsbyr_idx ON direct_sale_item(buyer_id);
CREATE INDEX autag_idx ON auction_item_tags(tag);
CREATE INDEX dstag_idx ON direct_sale_item_tags(tag);
```

Design Notes: We used a *node.js* framework with the following node-modules: *crypto* (for password hashing), *body-parser, connect-flash(for displaying flash messages for example to display "Invalid Username/Password" during Login), npm cookie (for storing the details of the current user as a cookie in the browser), ejs* (for our web page templates), *express.* We used *postgres* for database management and particularly the PostGis package for geospatial data.

TEST RESULTS

Test	Inputs	Outputs and backend changes	Test Procedure	Test Result and Remarks
1	Sign up with valid name, email-id, password, re-enter password, phone number, location	Redirected to login screen New row is inserted into person table	Database is queried to check the new entry in person relation	**Pass**
2	Sign up with invalid value or no value for at least one of the following: name, email-id, password,re-enter password, phone number, location	Refreshed version of the same page with an error message if password and re-enter password do not match, email is taken and in other cases message beside the incorrect field	Tested with several invalid values- already existing email-id, simple text in the email field, re-enter password different from password, a string as phone number	**Pass**
3	Login with valid email, password	Logged in and Home screen is loaded	Tested with valid email, password	**Pass**
4	Login with invalid email or password	Invalid details message is displayed and same page is refreshed	Tested with non-existent email, valid email with incorrect password	**Pass**
5	Log out button is clicked	Logged out and login screen is loaded	Tested by clicking on logout	**Pass**
6	Go to home screen button is clicked	Home screen is loaded	Tested by clicking on Home	**Pass**

7	Add money button is clicked and non-negative integer is entered	Balance updated message is displayed and redirected to the same screen balance of person is updated	Tested with some non-negative values and verified by the value of balance shown	**Pass**
8	Add money button is clicked and negative integer is entered	Invalid amount is entered message is displayed	Tested with some negative values and checked for error message and no change in balance	**Pass**
9	Withdraw Money button is clicked and amount is between 0 and balance - (on_hold_balance)	Withdraw successful message is displayed balance of person is updated	Tested with some valid values and verified by the value of balance shown	**Pass**
10	Withdraw Money button is clicked and above condition is not satisfied	Invalid amount is entered message is displayed	Tested with negative value, value above limit and checked for error message and no change in balance	**Pass**
11	Add new product for sale with valid details	Sale is added into direct_sale_item or auction_item table, relevant message is added to auction_item_messages or direct_sale_item_messages and redirected to home screen	Tested with valid values in relevant fields and verified by checking the new rows in database (and also by using view my sales, view my messages)	**Partially failed** (i)Sometimes, messages may not be generated and the app crashes, but the item gets added successfully
12	Add new product for sale with invalid details	Message is displayed at the incorrect field	Tested by giving invalid or no values in fields - name, price, quantity, date for auction_item and checking for validity	**Pass**

			message	
13	Search for an item with search_key and tags	Redirected to search screen with relevant search results	Tested with several combinations of search text and chosen tags	**Pass**
14	Select a product	Redirect to product details page	Tested upon both direct sale product and auction product	**Pass**
15	Place a direct order with sufficient funds	Order is placed and status of item is updated to sold and buyer id is set, new messages are added to direct_sale_messages	Tested by checking the new state of the item in the database (and also by viewing the product from buyer, seller), new messages in the database (and by view messages from buyer, seller)	**Pass**
16	Place a direct order with Insufficient funds	Insufficient funds message is displayed	Tested by placing order for an item with price more than (balance - on hold balance) and checking for error	**Pass**
17	Place a bid with bid value greater than or equal to the base price and in simple mode	If he is the best bidder it is displayed on screen and updated in the auction_item table and new row inserted into bid table and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by placing a bid on an item with no biddings, an item with biddings and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**

18	Place a bid with bid value greater than or equal to the base price and bid limit greater than or equal to bid value in auto mode	If he is the best bidder it is displayed on screen and updated in the auction_item table and new row inserted into bid table and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by placing a bid on an item with no biddings, an item with biddings and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**
19	Place a bid with any of the above conditions not satisfied	Suitable error message is displayed	Tested by placing bid less than base price, bid limit less than base price, bid limit less than bid (latter two only for auto bidding) and check for error message	**Pass**
20	Place a bid with insufficient funds	Insufficient funds message is displayed	Tested by placing bid more than available balance, bid limit more than available balance (inclusive of delivery cost) and checking for error message	**Pass**

21	Update bid with new bid value greater than or equal to the previous bid value(both in simple mode)	If he is the best bidder it is displayed on screen and updated in the auction_item table and bid table entry is updated with new values and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by updating bid and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**
22	Update bid with new bid value greater than or equal to the previous bid value and bid limit greater than or equal to the previous bid limit (both in auto mode)	If he is the best bidder it is displayed on screen and updated in the auction_item table and bid table entry is updated with new values and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by updating bid and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**
23	Update bid with new bid value greater than or equal to the previous bid value and bid limit greater than or equal to the new bid value(first in simple and second in auto mode)	If he is the best bidder it is displayed on screen and updated in the auction_item table and bid table entry is updated with new values and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by updating bid and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**

24	Update bid with new bid value greater than or equal to the previous bid limit(first in auto and second in simple)	If he is the best bidder it is displayed on screen and updated in the auction_item table and bid table entry is updated with new values and bid values of other buyers may be changed and relevant message is added to auction_item_messages and on hold amount of buyer is updated	Tested by updating a bid and verifying by checking the new version of database (and by viewing the bids, item, messages), checking the new on hold balance of the buyer	**Pass**
25	Update bid with any of the above conditions failing	Suitable error message is displayed	Tested for invalid versions of the above cases and checked for error message	**Pass**
26	Update a bid with insufficient funds	Insufficient funds message is displayed	Tested by updating with bid more than available balance, bid limit more than available balance (inclusive of delivery cost) and checking for error message	**Pass**
27	Delete product for direct/auction item is clicked	Status of item is set to closed and in case of auction item the on hold balances of all the rejected buyers is updated and redirected to home screen and auction_message is added with "product deleted message" for seller and bid status of bidders is changed to rejected/removed	Tested by deleting auction item and checking the on hold balances of bidders, status of the item (by view product), messages (by view messages) and direct sale item by the status of the item (by view product), messages (by view messages)	**Pass**

28	View bids button is clicked	Redirected to view my bids page	Tested by clicking	**Pass**
29	View sales button is clicked	Redirected to view my sales page	Tested by clicking	**Pass**
30	View my orders button is clicked	Redirected to view my purchases screen	Tested by clicking	**Pass**
31	View my messages button is clicked	Redirected to messages screen	Tested by clicking	**Pass**
32	Close bidding button is clicked by seller and current best bidder is not empty	Status of product is changed to auctioned, on-hold balance of rejected buyers is updated, auction_messages is added for buyer and seller with bidding closed message and bid status of rejected buyers is changed to rejected/removed and bid status of best bidder is changed to accepted	Tested by closing the bidding of an auction item with bidders and checking the on-hold balance of rejected bidders, checking for item status, messages	**Pass**
33	Close bidding button is clicked by seller but no bid is present(current best bidder is empty)	No bid is present message is displayed	Tested by closing bidding for an auction item with no bidders and checking for error message	**Pass**
34	Update Delivery status button is clicked by the seller	Status of the product is updated in the direct/auction table and status is changed on the screen and messages are sent to buyer and seller with updated status	Tested by updating the status of the item to several statuses for both auction item, direct sale item and checking the updated status, checking messages	**Pass**

35	Buyer confirms delivery	Status of the product is updated to delivered in the direct/auction table and status is changed on the screen and messages are sent to buyer and seller saying product delivered, on hold balance and balance of buyer are reduced and balance of seller is increased	Tested by updating the status of the item to delivered for both auction item, direct sale item and checking the updated status, new balances of buyer and seller, on hold balance of buyer, checking messages	**Pass**

Note: Querying the database mentioned in some of the test procedures can be performed by either querying through pgadmin or postgresql interface.

REPO

eMercado