FriedCoke Auction Website APIs

FrontEnd Functions:

- Create User
    POST /api/user
    int addUser()
-Delete User
    DELETE /api/user/<id>
    int deleteUser()
-Suspend User
    PUT /api/user/<id>
    int updateUser()
-Login
    POST /api/login
    int login()
-Logout
    POST /api/user/<id>/logout
    int logout()
-See active actions
    GET /api/auction?status=active
    List<Auction> getActiveAuctions()
-Add Item
    POST /api/auction
    addAuction()
-Bid on Item
    PUT /api/auction/<id>
    int bidAuction()
-Remove Item
    DELETE /api/auction/<id>
    int deleteItem()
-Purchase Item
    PUT /api/auction/<auctionId>/purchase/<userId>
    int purchaseAuction()

Database Functions:

- Store Data in a persistent manner. (If the user shuts down your app, then powers it back up, all the data from the previous instance should still be there)
- Retrieval of stored data by both Frontend and Backend services. (For example, retrieval of all auction items for a specific user)
- Ability to perform all CRUD operations on your data from both the Frontend and Backend services.
- Database should scale as data are added or removed.

Admin Functions:

-Stop an auction early

PUT /api/auction
int closeAuction()

-Remove and block a user

PUT /api/user/<id>/block
int blockUser()

-Add, modify, or remove categories

POST /api/category
int addCategory()
PUT /api/category
int updateCategory()
DELETE /api/category
int deleteCategory()

-View all items that have been flagged by users

GET /api/auction?flag=true
List<Auction> getFlaggedAuctions()

-View all auctions currently in progress, and include sorting capability so that auctions ending soonest can be displayed first

GET /api/auction?status=active
List<auction> getActiveAuctions()

-Examine metrics for closed auctions in a given timeframe(last day, week, month, etc)

GET /api/auction?status=closed
List<auction> getFinishedAuctions()

-Examine emails that are received by customer support, and respond to these emails within the admin functionality

GET /api/email?receiver=admin
List<Email> getEmails(receiver)

POST /api/email
int sendEmail()

Auction Functions:

-Allow listing of items for bidding

GET /api/auction?status=active
List<auction> getActiveAuctions()

-Start the auction when the current time matches the start time defined by the user

Solved by while loop in auction microservice

-Allowing auction window to be set by the bidder, and begin countdown to the end of the bidding window once auction begins

POST /api/auction
int addAuction()

-Allow bid to be placed, and increment bid amount as users enter new bids

PUT /api/auction/<id>
int bidAuction()

-Allow item to be categorized by user

POST /api/auction
int addAuction()

-Allow search of items on the site by keyword, or item category
    GET /api/auction?keyword=xx
    List<auction> getAuctionsByKeyword(keyword)
-Allow item to be placed on a watchlist for a user, that includes parameters defined by the user(i.e. Ray-Ban sunglasses less than $100 in starting price)
    POST /api/user/<id>/watchlist
    Int addItemToWatchlist()
-Send a email notification to user when an item on their watchlist appears matching their criteria
    Solved by while loop in auction microservice
-Allow multiple bids to be placed at once by different users
    PUT /api/auction/<id>
    int bidAuction()
-Alert seller when their item has been bid on with an email
    Solved by while loop in auction microservice
-Alert buyer via email when someone has placed a higher bid on the item they had bid current high bid on
    Solved by while loop in auction microservice
-Implementation of a shopping cart that will store multiple items in it while the user shops on the site
    GET /api/user/<id>/cart
    List<auctionId> getShoppingCart(user)
    POST /api/user/<id>/cart
    Int addItemToShoppingCart()
-Allow item to be place in the shopping cart if the Buy Now feature is selected
    POST /api/auction/<auctionId>/buynow/<userId>
    Int addItemToShoppingCart() + int buyNow()
-Place item in a user's cart if they have the winning bid when the auction expires
    POST /api/user/<id>/cart
    Int addItemToShoppingCart()
-Allow a user to checkout from their cart once there are items in it
    POST /api/user/<id>/checkout
    Int checkout()
-Alert both seller and bidders when on predetermined time setting, 1 day before bidding ends, 1 hour before bidding ends, etc
    Solved by while loop in auction microservice
-Remove auction once bidding is complete and user checkouts out
    PUT /api/auction/<id>/terminate
    Int terminateAuction()

User Functions:

-Create a new user(user should have the ability to place bids on a item(s) or place an item for sale, or both)
    POST /api/user
    Int addUser()
-Update a user's information
    PUT /api/user/<id>/update
    Int updateUser()
-Delete a user
    DELETE /api/user/<id>
    Int deleteUser()
-Suspend an account

PUT /api/user/<id>/suspend
Int suspendUser()

-List an item for auction(item should have start price, start time, quantity, time when the auction expires, shipping costs, buy now feature if applicable, item description, seller rating)
POST /api/auction
Int addAuction()

-Update an item properties, including quantity, description, shipping costs, buy now feature
PUT /api/auction/<id>
Int updateAuction()

-Flag an item as inappropriate or counterfeit
PUT /api/auction/<id>
Int flagAuction()

-Categorize an item based on existing categories or create an new category if needed
Do this when adding a new auction

-Delete an item if there are no bids on it, but do not allow an item to be deleted if there are bids on it
Check if there is a highest bidder before deleting an auction

-Bid on an item, and update that bid if another user outbids that user
Covered

-See a list of all items that are currently being bid on by that user
GET /api/user/<id>/biddinglist
Int getBiddingList(user)

-Add item to cart directly via the Buy Now functionality
Covered

-Checkout of the auction once winning by selecting checkout from the cart
Covered