CSC 322 Group Project - Report 2
Misael Perez
Carlos Rodriguez
Md Omit

Project Pseudo Code

Visitor Actions (V)

Browse Listings
Method: browseListings
Input: none
Output: List of available items/services with details
Functionality:
- Retrieve items/services available for sale or rent.
- Display information including item details, owner, price, and type (sale or rent).

Submit Application
Method: submitApplication(visitorId)
Input: visitorId
Output: Success or Failure message
Functionality:
- Generate a random arithmetic question.
- Verify visitor response.
- If correct, submit the application to Super-user for approval.
- Return success or failure of application submission.

User Actions (U)

Deposit Funds
Method: depositFunds(userId, amount)
Input: userId, amount
Output: Updated balance or error message
Functionality:
- Validate amount is positive.
- Add amount to user’s account balance.
- Return updated balance or error if invalid.

Withdraw Funds
Method: withdrawFunds(userId, amount)
Input: userId, amount
Output: Updated balance or error message
Functionality:
- Check if user has sufficient balance.
- Deduct amount from user’s account balance.
- Return updated balance or error if insufficient funds.

List Item/Service
Method: listItem(userId, itemDetails, price, listingType)
Input: userId, itemDetails, price, listingType (sale or rent)
Output: Listing confirmation or error message
Functionality:
- Validate item details and price.
- Save item to the listings database.
- Return confirmation of listing or error if invalid input.

Place Bid
Method: placeBid(userId, itemId, bidAmount, deadline)
Input: userId, itemId, bidAmount, deadline
Output: Success or Failure message
Functionality:
- Check if user has sufficient funds.
- Submit bid with specified amount and deadline.
- Return success or failure based on bid validity.

Complete Transaction
Method: completeTransaction(ownerId, buyerId, itemId)
Input: ownerId, buyerId, itemId
Output: Transaction success message or error
Functionality:
- Validate buyer funds and ownership of item.
- Transfer item from owner to buyer.
- Transfer funds from buyer's account to owner's account.
- Remove item from listings.
- Return transaction success message or error if failed.

Rate Transaction
Method: rateTransaction(raterId, transactionId, rating)
Input: raterId, transactionId, rating (1 to 5)
Output: Rating success message or error
Functionality:
- Check if rating is within acceptable range (1-5).
- Store rating anonymously for transaction.
- Update user's rating average.
- Return success or error message based on validity.

File Complaint
Method: fileComplaint(userId, targetUserId, transactionId, complaintDetails)
Input: userId, targetUserId, transactionId, complaintDetails
Output: Success or failure message

Functionality:
- Verify complaint validity (within past transactions).
- Record complaint details for Super-user review.
- Return confirmation or error if invalid.

Super Actions (S)

Approve Application
Method: approveApplication(visitorId)
Input: visitorId
Output: Approval or denial message
Functionality:
- Review visitor’s application.
- Approve or deny based on criteria.
- Convert approved Visitor to User.
- Return confirmation of approval or denial.

Handle Complaint
Method: handleComplaint(superUserId, complaintId, action)
Input: superUserId, complaintId, action (e.g., warn, suspend, no action)
Output: Resolution message
Functionality:
- Review complaint details.
- Decide on action (warn, suspend user, etc.).
- Execute selected action.
- Record action in system and return resolution message.

VIP Status Management

Check Eligibility
Method: checkVipEligibility(userId)
Input: userId
Output: Eligibility status (True/False)
Functionality:
- Check if user meets VIP criteria:
  - Balance > $5,000.
  - More than 5 completed transactions.
  - No active complaints.
- Return eligibility status.

Apply VIP Discount
Method: applyVipDiscount(userId, transactionAmount)
Input: userId, transactionAmount
Output: Discounted amount or original amount if not VIP
Functionality:
- Check if user is VIP.

- If VIP, apply 10% discount to transactionAmount.
- Return discounted amount.

Revoke VIP Status
Method: revokeVipStatus(userId)
Input: userId
Output: Success message
Functionality:
- Check if user no longer meets VIP criteria.
- Revoke VIP status and revert to regular User.
- Return success message if VIP status revoked.

User Ratings & Suspension

Evaluate Rating
Method: evaluateUserRating(userId)
Input: userId
Output: Suspension status (suspended or active)
Functionality:
- Calculate average rating for user.
- Suspend if:
  - Average rating < 2 with at least 3 ratings.
  - Unusual rating behavior (too low or high).
- Return suspension status.

Suspension Handler
Method: handleRepeatedSuspensions(userId)
Input: userId
Output: Suspension status (permanent or temporary)
Functionality:
- Check user's suspension history.
- Permanently remove user if they've been suspended three times.
- Return status of user account (active or removed).

User Interface & Dashboard

Method: displayUserDashboard(userId)
Input: userId
Output: Personalized user dashboard with relevant data
Functionality:
- Retrieve user-specific information (transaction history, ratings, account balance).
- Display personalized details on GUI.
- Include VIP status and any pending actions (bids, complaints).

Unique Functionality

Price Notifier
Method: setPriceAlert(userId, itemId, targetPrice)
Input: userId, itemId, targetPrice
Output: Confirmation of alert or error
Functionality:
- Save alert with user's ID, item ID, and target price.
- Monitor item prices.
- If item's price <= target price, notify user via dashboard or email.
- Return alert set confirmation or error if input is invalid.

Wishlist
Method: addItemToWishlist(userId, itemId)
Input: userId, itemId
Output: Wishlist update confirmation
Functionality:
- Add item ID to user's wishlist.
- Monitor item for availability and price changes.
- Notify user if item meets criteria.
- Return confirmation of addition.

Meeting Information

We held one of our most important meetings last week after the class was over. In this meeting, we discussed the project scope as well as how we were each going to tackle the diagrams at our disposal. We ran into a bit of an issue since two of our original group members backed out of the project at a last minute notice. However, once the group members and I held our meeting, we discussed each others skillset and realized that we could finish the project in its entirety. As such, since the last time we met we were able to establish a consistent GitHub repository to upload our work and have that as our main workspace, and were able to clean up the Discord channel that we are using as a means of communication. Our configurations and workspaces have been revamped to allow for ease of communication as we transition into the coding phase of our final project!

Our main concerns of the teamwork have mainly been because of the sudden exit of two of our teammates, one of which was the one who created our original discord. As such I, Carlos Rodriguez, have assumed ownership of the server. One of the other concerns is definitely about time management. Given that we all have other classes that we need to study for, it is going to be interesting and difficult in regards to how we manage our time. Flexibility in the schedule is an issue that pertains to this as well. Team size is another issue, since now we only have one person fully dedicated to the BackEnd while the other teammates are FrontEnd. Of course we can step in and review the code, but having such a small team is another project in it of itself.

GitHub Information

GitHub Repo: Chromium99/CSC322GroupProject