Team H
(Abdul Muswara, Arif Manawer, Mohammad Kabir, Mohammed A. Al-Muqsit)

Project E-Bidding

Phase 2: Design Report

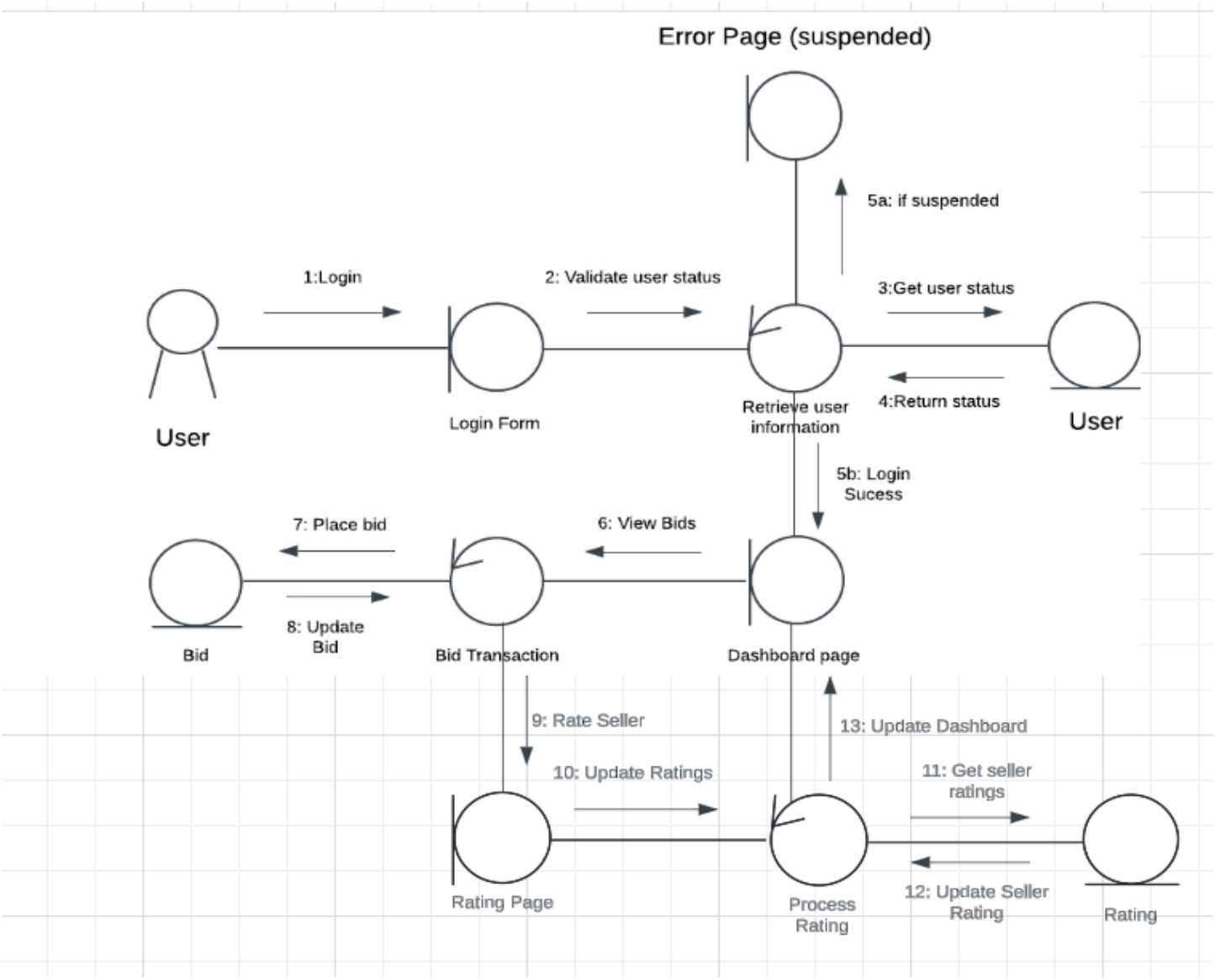
Date	Version	Description	Author
10/17/2024	1.0	Initial Draft	Abdul Muswara, Arif Manawer, Mohammad Kabir, Mohammed A. Al-Muqsit, Ibrahim Rahat
11/12/2024	2.0	Design Report	Abdul Muswara, Arif Manawer, Mohammad Kabir, Mohammed A. Al-Muqsit

Table of Contents

---

1. Introduction
   an overall picture of the system using collaboration class diagram

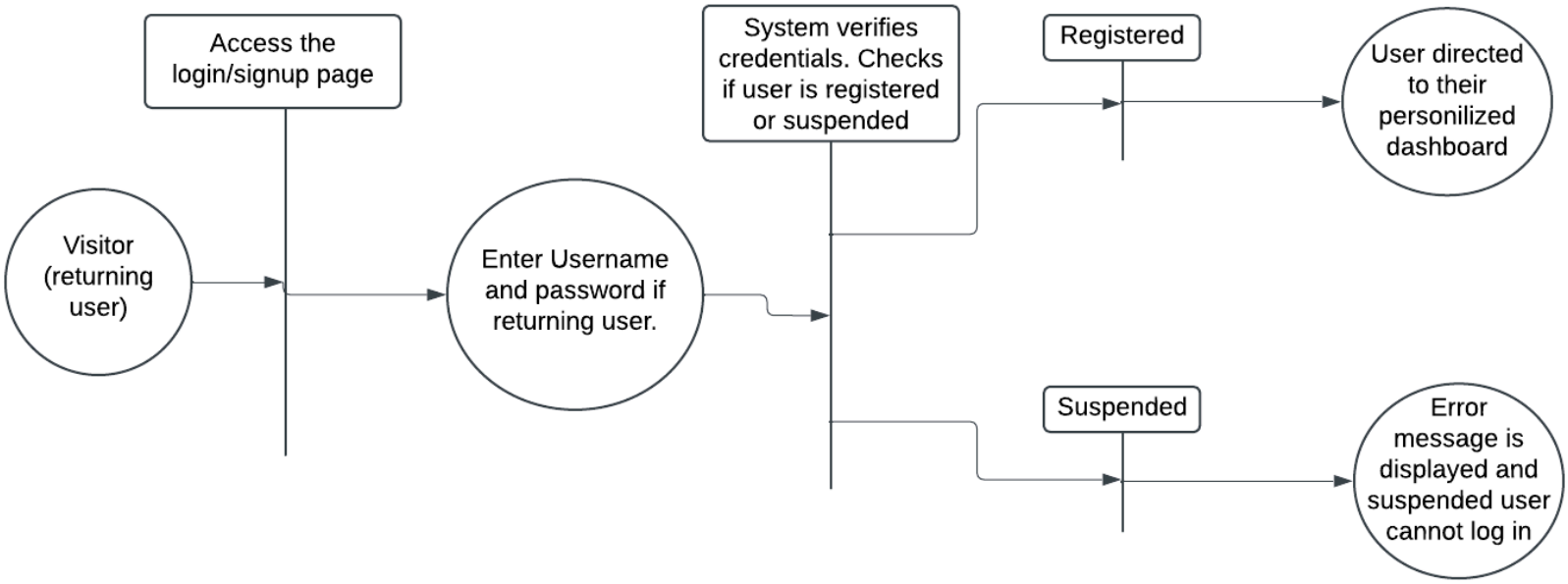
This is the overall picture of the system with the assumption that the user exists already. Generally, a user will login and if they are suspended they will not be allowed to login.



2. All use cases
   o Scenarios for each use case: normal AND exceptional scenarios
   o Collaboration or sequence class diagram for each use case,
choose 3 or more use cases: draw the Petri-nets instead of class diagrams

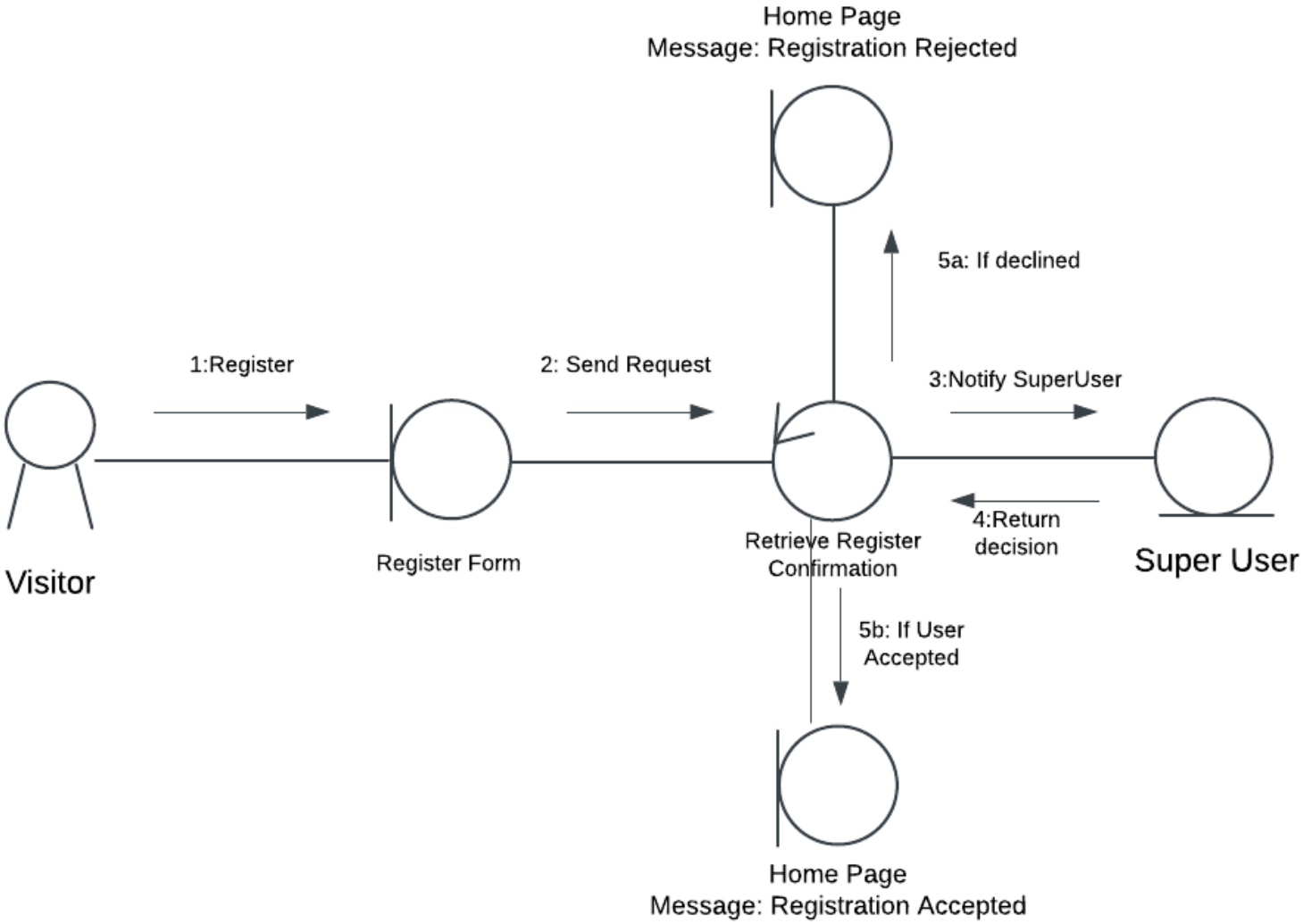
2.1 User Authentication

This use case involves a user attempting to log in to the system. If the user is new, they are prompted to create an account. Suspended users encounter an error message and are prevented from logging in. A successful login leads the user to their dashboard, where they can access various functionalities, such as bidding on items and rating other users. This process ensures that only authorized and active users can interact with the system.



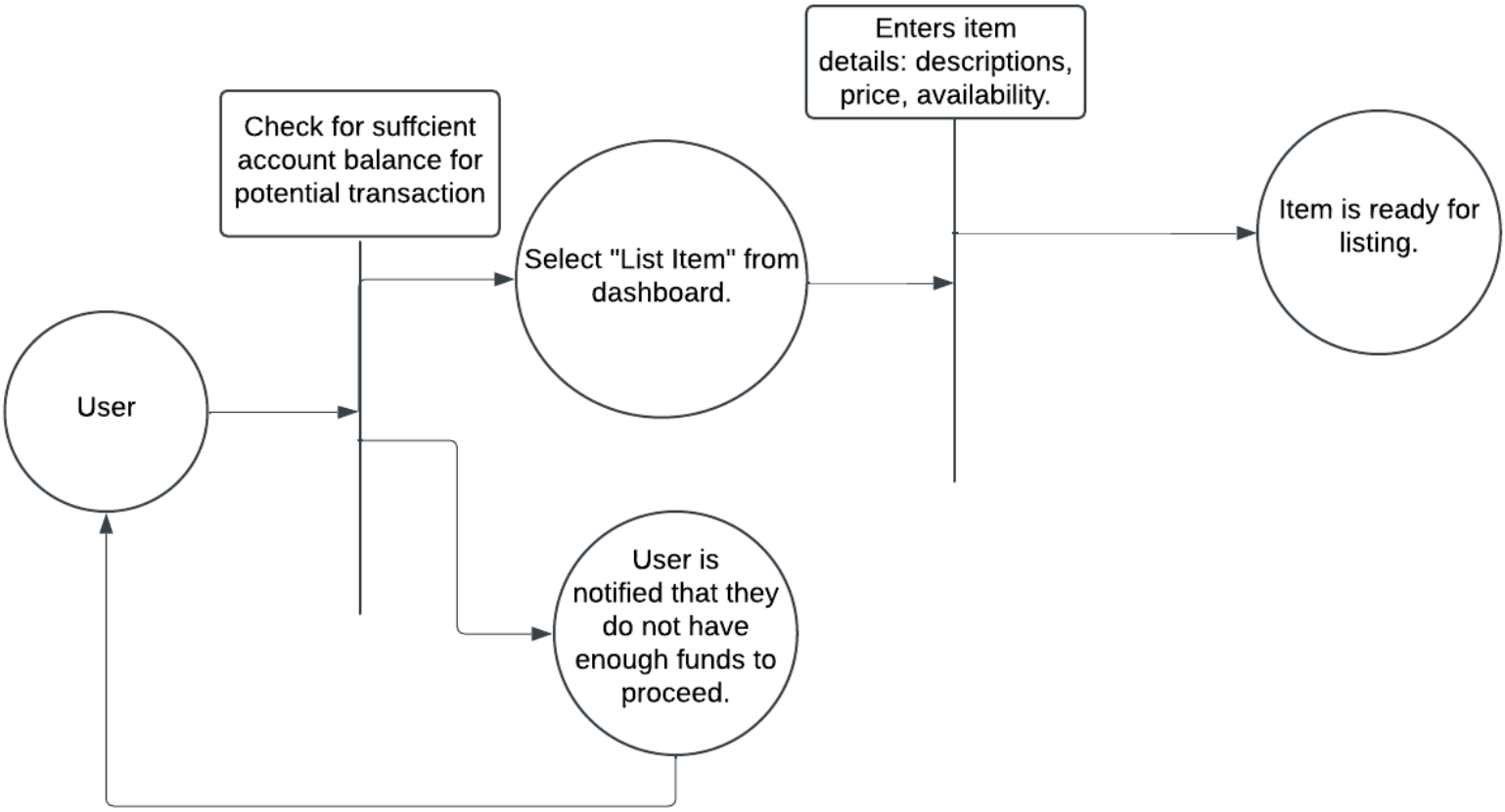
2.2 Applying to Become a User

New users must complete an application to register, which includes passing a verification step to confirm they are not bots. Once the application is successful, the user is directed to the homepage, receiving confirmation of their account creation. If the registration fails, the user is notified and redirected to a page with details on why the application was rejected. This step helps maintain the integrity and security of the platform by allowing only verified users.



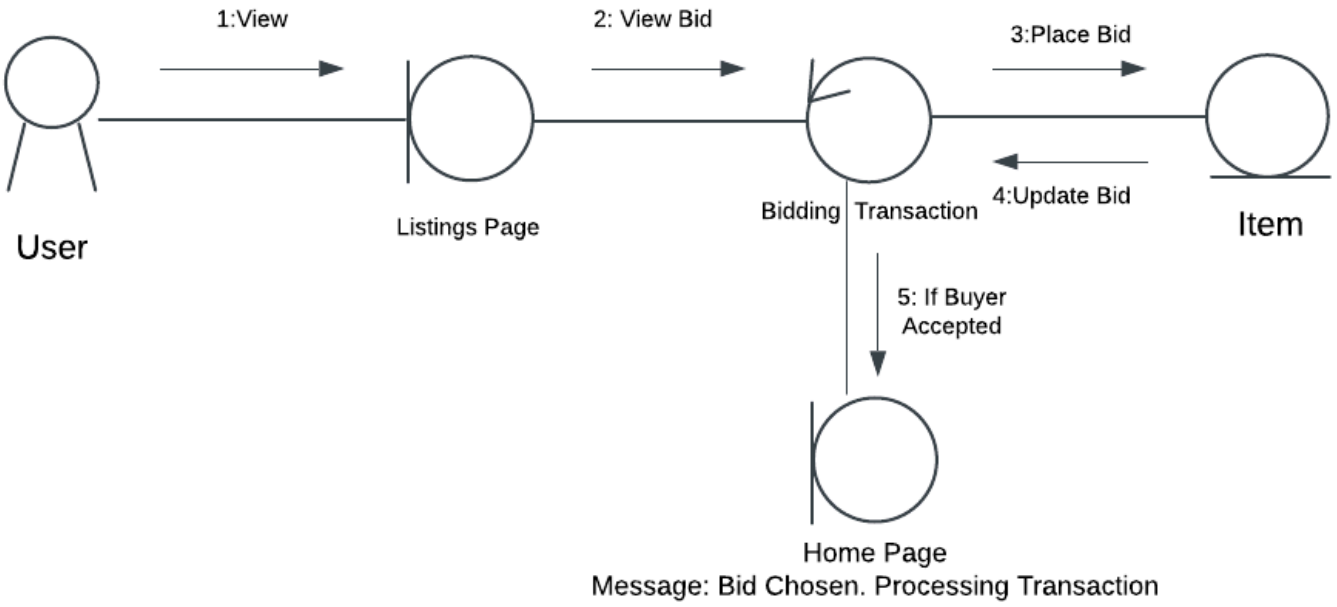
2.3 Listing an Item

Users with adequate funds can list items for bidding. They specify the item's details, such as description, minimum bid, and maximum bid. The system then verifies the information, and once complete, the item becomes available for bidding by other users. This use case ensures that only eligible items and users can participate in the listing process, maintaining a reliable marketplace.



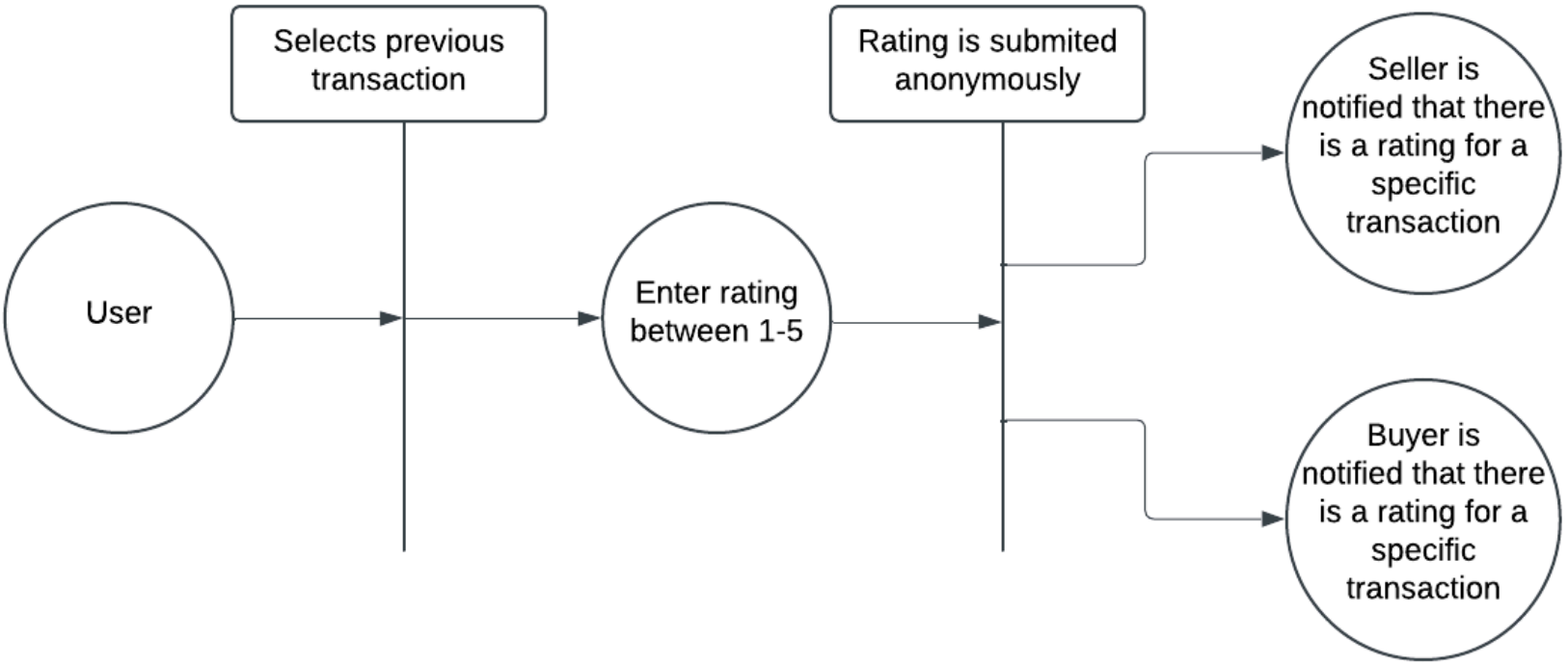
2.4 Bidding on an Item

Users can view available listings and place bids on items they are interested in. Upon placing a bid, the item's status and bid details are updated accordingly. If the user's bid meets the current requirements, the transaction proceeds and updates, and the user is taken back to the homepage. This process facilitates competitive bidding while ensuring smooth navigation for users.



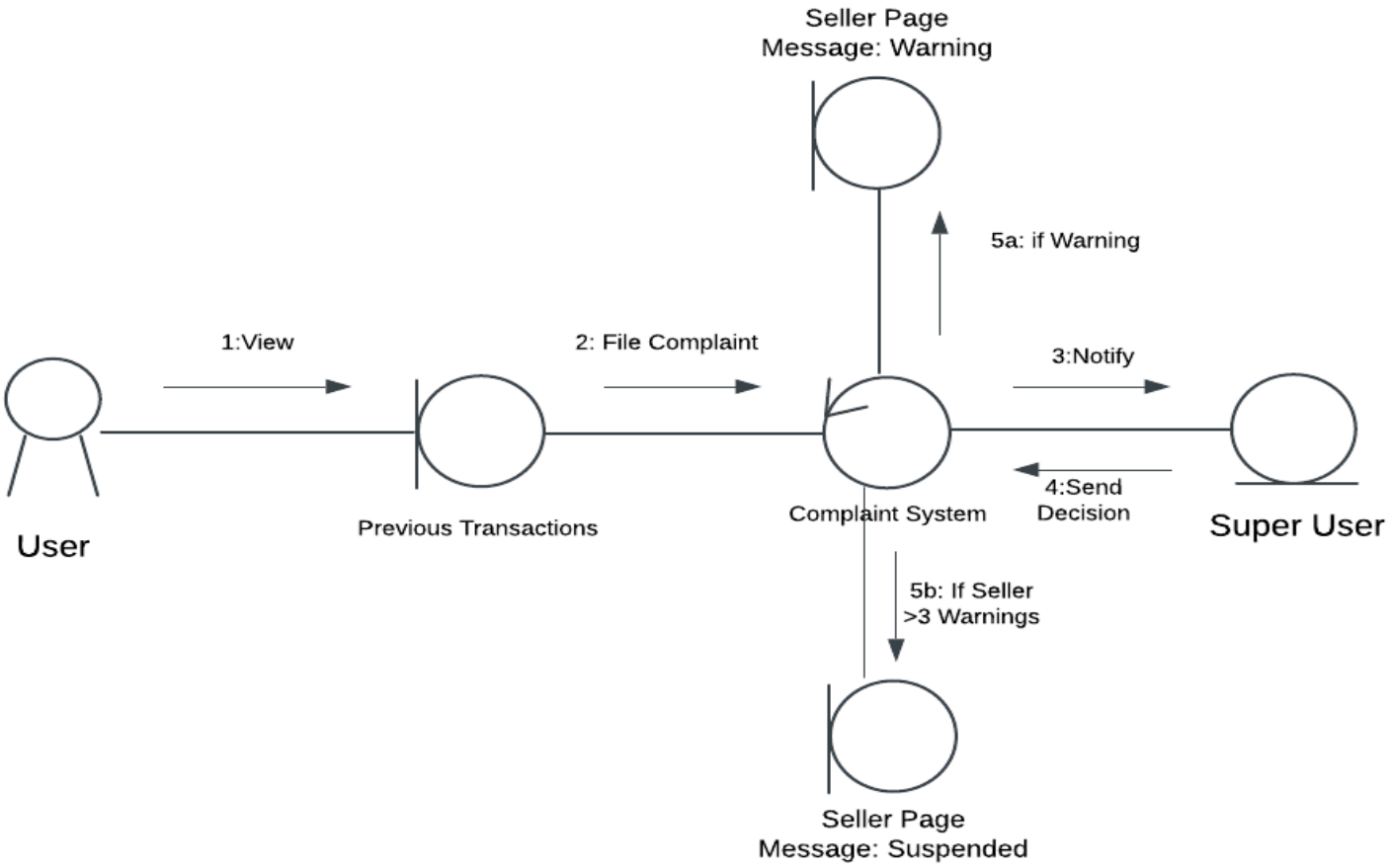
2.5 Rating a User

After completing a transaction, users have the option to rate each other. To do so, they select the relevant transaction and provide a rating that remains anonymous. Only users involved in a transaction can provide ratings, ensuring authenticity and preventing abuse. This feedback system contributes to trust and accountability within the platform.



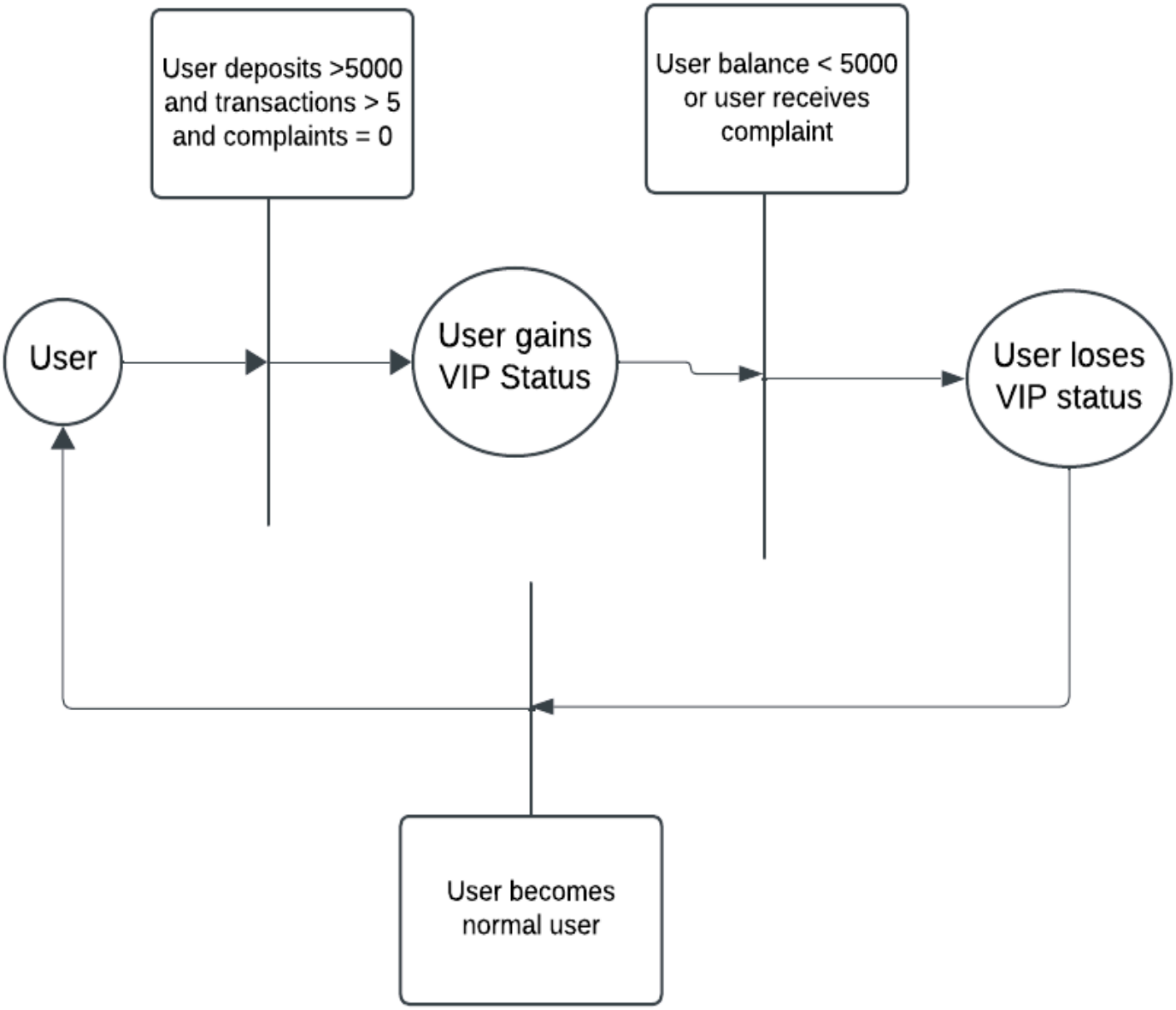
2.6 Handling Complaints

Users may file complaints regarding previous transactions. They must select the relevant transaction, after which a super user is notified to review the complaint and decide on a course of action. This mechanism allows the platform to address issues fairly and efficiently, improving user experience and maintaining a safe marketplace.



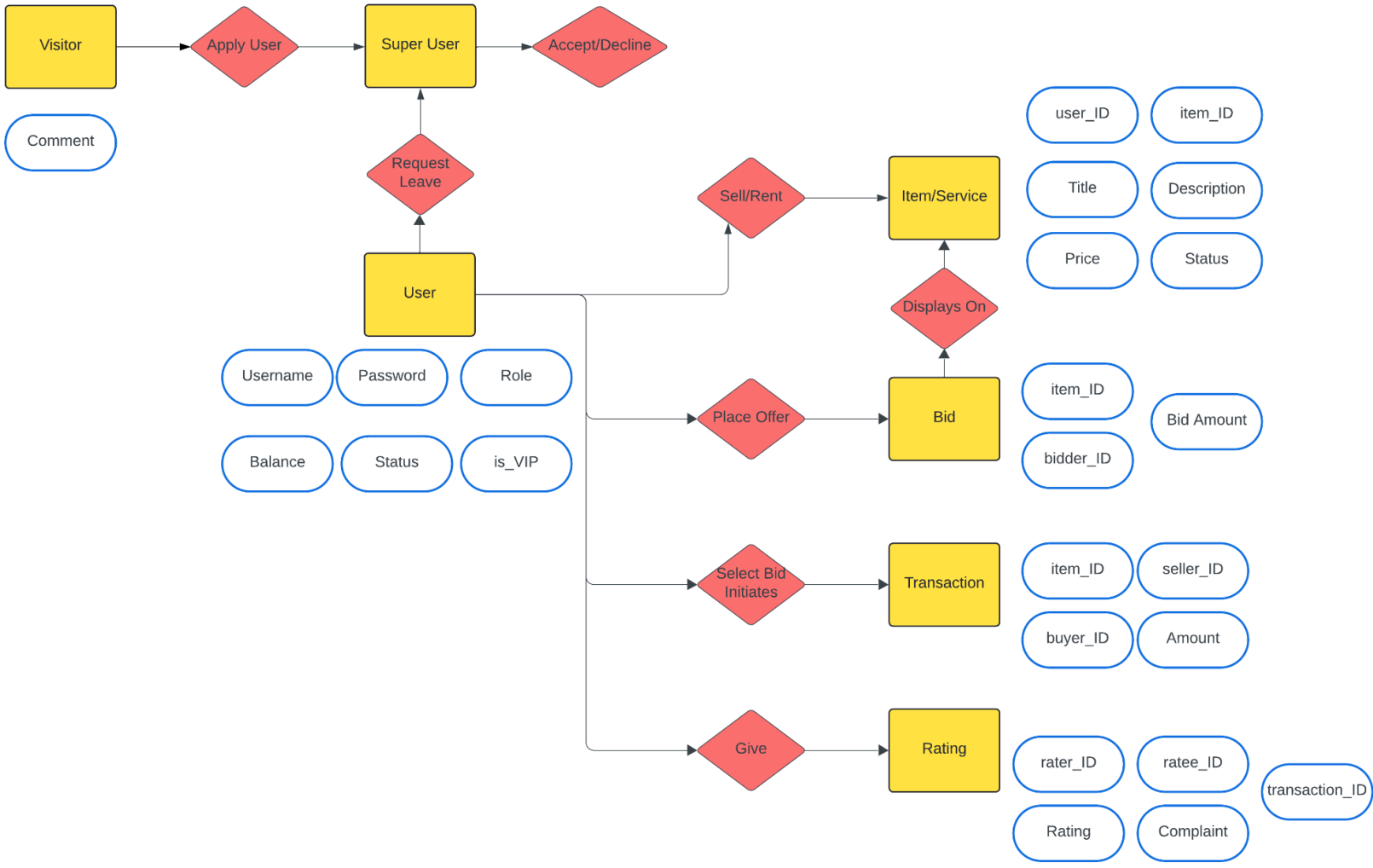
2.7 Managing VIP Status

To achieve VIP status, a user must meet specific conditions, such as maintaining a balance above $5,000. VIP users enjoy privileges like access to exclusive bidding rooms but face strict compliance standards; violations can result in their VIP status being downgraded to regular user status. This use case incentivizes responsible platform usage and rewards high-engagement users with enhanced features.



3. E-R diagram

For the entire system demonstrates the entities of the systems and how the user interacts with different parts and their attributes, the Entity-Relation Diagram below perfectly describes the entire process.



4. Detailed design:

for EVERY method use pseudo-code to delineate the input/output and main functionalities

RatingSeller: After the transaction is done, the buyer/renter can rate the owner: from 1 - worst to 5 - best. U's do not know who rates them so they cannot retaliate or praise back. No one else can rate. A U can directly complain to S about the U s /he bought/rented items

```
If (incomplete transaction || not buyer)
        Error Message: "Cannot rate the seller"

User is prompted to rate the item/service of the seller 1-5
If (rating not within range 1-5)
        Error Message: "Rating Invalid"
Add Item.rating to seller's profile
```

RatingBuyer: The owner can rate the renter after the transaction. U's do not know who rates them so they cannot retaliate or praise back. No one else can rate. A U can directly complain to S about the U s /he bought/rented items

```
If (incomplete transaction || not seller)
        Error Message: "Cannot rate the buyer"

User is prompted to rate the item/service of the buyer 1-5
If (rating not within range 1-5)
        Error Message: "Rating Invalid"
Add Listing.rating to buyer's profile
```

Write Comment on Listings:

```
If (listing is not found)
        Error Message: "Listing does not exist"
Else
        commentText added to Listing.comments
```

Apply for User: Including questions to be answered by visitors to confirm they are not robots.

```
User is prompted to answer human-verification question
If (fail)
        Error Message: Registration Failed
Else
```

Send all user information + request to Super User

RegistrationApproval:

```
if (user.role == 'Super')
    If (SU approves)
        User is registered into system
    Else
        Registration Declined message
```

LeaveRequestApproval:

```
if (user.role == 'Super')
    If (SU approves)
        User is taken out of system
    Else
        Suspension Declined message
```

AddMoney:

```
Balance = user.balance
Get Deposit_amount from user input

Balance = Balance + Deposit_amount
Deposit_amount is taken from bank account
```

WithdrawMoney:

```
Balance = user.balance
Get Withdraw_amount from user input

If   (Withdraw_amount < Balance)
        Balance = Balance - Withdraw_amount
        Withdraw_amount is sent to bank account
Else
        Return error message
```

AddingListing: can list any items/services s/he owns and the asking price for sale or rent can list any items/services s/he needs and the price range s/he can accept;

```
User inputs newListing's Title, Description, Min Bid, Max Bid

Items.append(newListing)
displayListing(newListing)
```

BiddingOnItems:

If (User has adequate balance for bid amount):
  User places a bid on the item
  Return the bid amount and User in the item's bid list
Else:
  Return "Insufficient balance to place bid" message

SellerChooseBuyer:

If (Bids exist for the item):
  Owner selects the preferred buyer (e.g., highest bid or other criteria)
  Set chosen buyer for the item
Else:
  Return "No bids available" message

Purchase:

If (Buyer has been chosen by the Seller):
  If (Buyer has sufficient balance for the item price):
    Deduct item price from Buyer's balance
    Add item price to Seller's balance
  Else:
    Return "Insufficient balance for purchase" message
Else:
  Return "No buyer selected" message

RemoveListingAfterPurchase:

If (Owner and Buyer agree to complete the purchase):
  Remove the item from the listing
  Return "Item successfully purchased and removed from listing" message
Else:
  Return "Agreement required to complete purchase" message

UserEvalulation(user): if < 2 by at least 3 U then suspend → unsuspend if pay

```
user.allRating #array that holds ratings from different users
user.poorRating = 0
for i in allRating:
        if user.allRating[i] < 2:
                user.poorRating += 1
```

```
            if user.poorRating >= 3:
                userSuspend(user)
                user.suspensionCount += 1
```

UserSuspend(user):

```
    if user.status == 'VIP':
        VIPSuspension(user)
    else:
        user.status = 'suspended'
```

UserUnsuspend(super, user):

```
    if user.fined == 50: #if they pay $50
        user.fined = 0
        user.status = 'normal'
        return
    super.setUserStatus('normal')
```

ForcedRemoval(user): 3 times suspension then removed

```
    if user.suspensionCount >= 3:
        user.status = 'removed'
    elif user.apply == True: #if user applied to be S then they are removed
        user.status = 'removed'
```

SetVIPStatus(user): 3 conditions in order to become a VIP

```
    if user.accountBalance > 5000:
        user.status = 'VIP'
    elif user.transactionCount > 5 && user.complaintsCount = 0:
        user.status = 'VIP'
```

VIPViolation(user): #if VIP user violates conditions they become a normal user

```
    if user.accountBalance <= 5000:
        user.status = 'normal'
    elif user.complaintsCount > 0:
        user.status = 'normal'
```

VIPSuspension(user):

```
    user.status = 'normal'
    user.suspensionCount += 1
```

ShowUserRecords:

If loggedIn():
    openDashboardPage()

OpenDashboardPage(user): #gets all using showXYZ information and displays them on the page

user.showBalance()
user.showStatus()
user.showSuspensionCount()
user.showItemsSold() #also shows ratings
user.showItemsBidding()

LiveBiddingRoom(item_id)
If (Listing does not have live bidding)
    Error Message: “Live Bidding is not available”

User is prompted to enter live bidding room

if (currentUser.status != ‘VIP’)
    Error Message: “Only VIPs”

Display current bids and timer

placeLiveBid(bidAmount, item_id):
If (currentUser.status is not "VIP")
    Error Message: "Access denied. Only VIPs can place live bids."

If (bidAmount < current highest bid)
    Error Message: "Bid must be higher than current highest bid."

Else
    current highest bid = bidAmount
    All participants can see the new highest bid and bidder
    Message: "Bid placed successfully"

endLiveBid(item_id):

If (live bidding session timer has ended for itemId)

Get the highest bid and winning user

Message the winner and item owner of the bid result

Process transaction for winning bid

Close bidding room

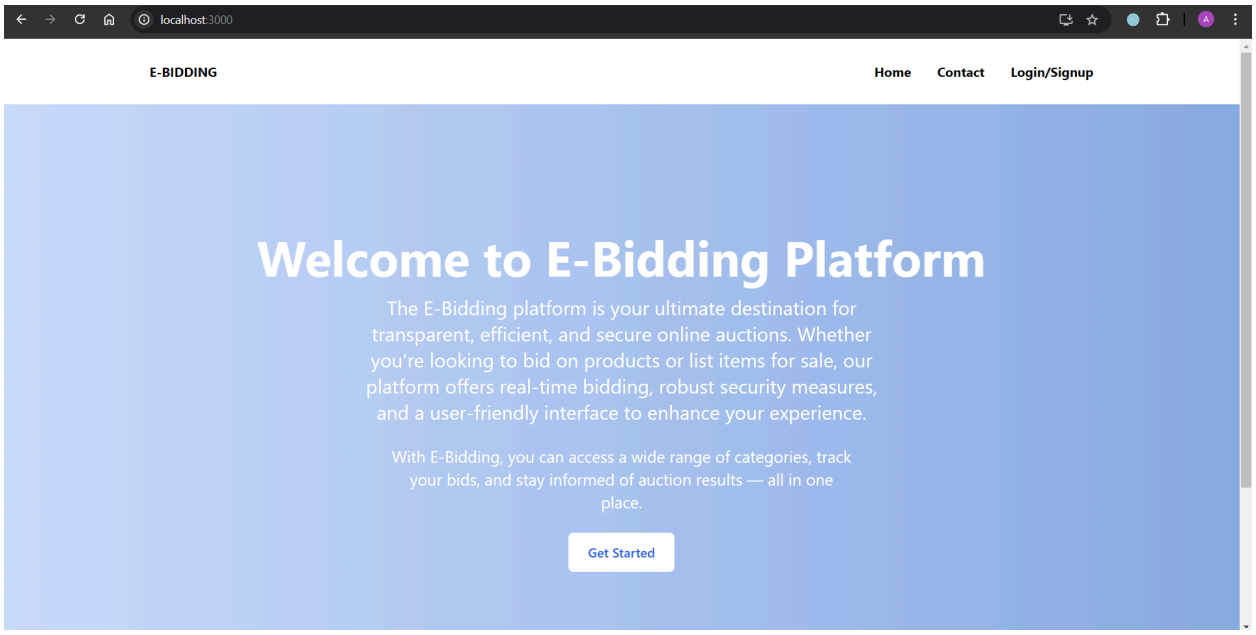
Return Message: "Live bidding session has ended."

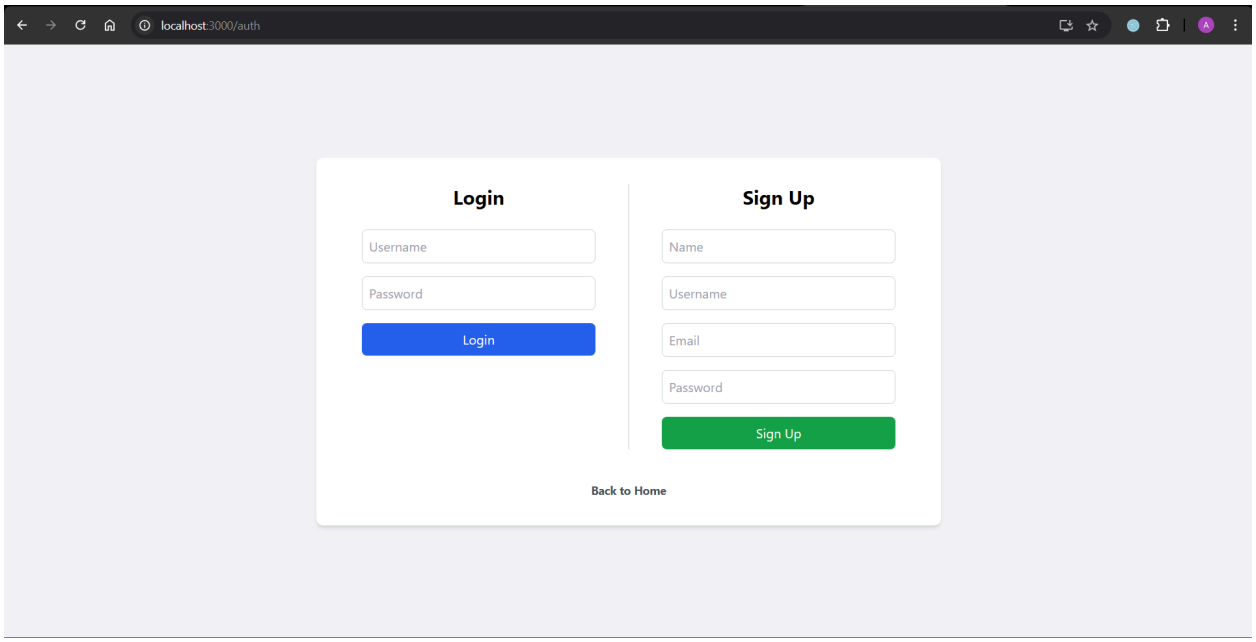
Else

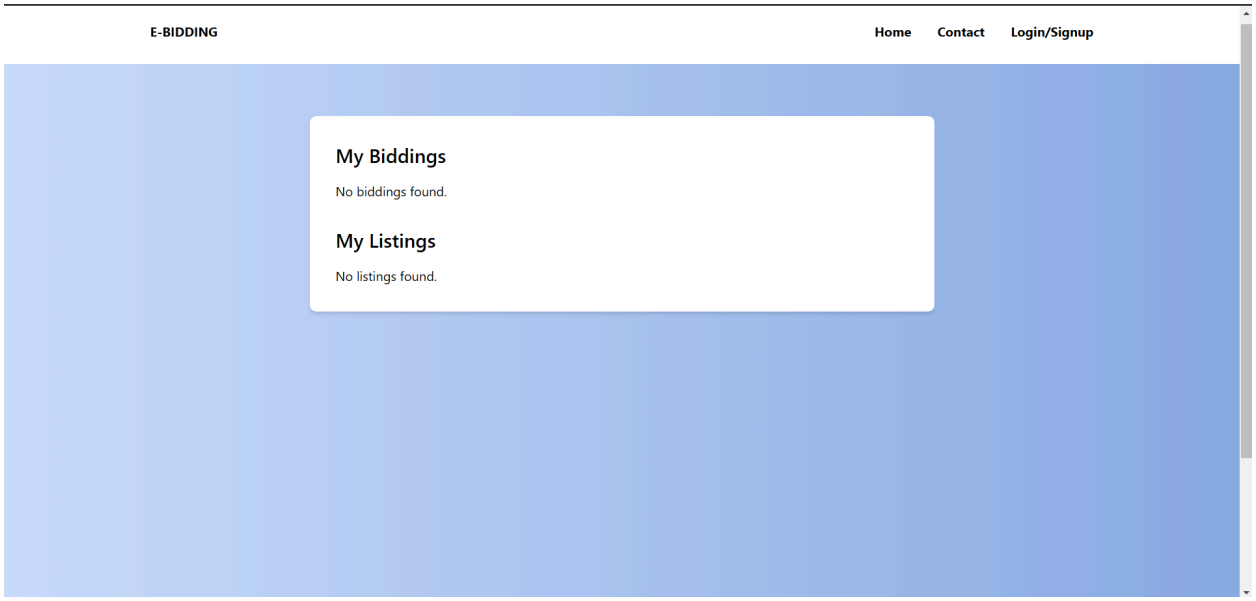
Return Message: "Live bidding session is still active."

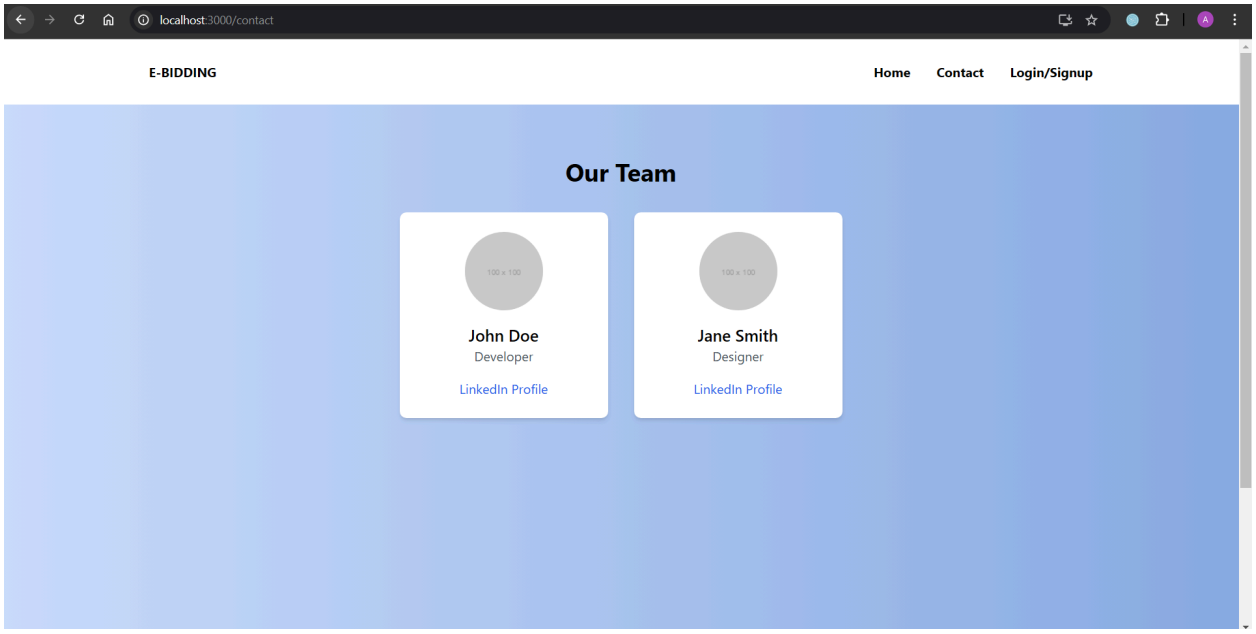
5. System screens:

demonstrate major GUI screens of the system and a sample prototype of one functionality of your own choice.

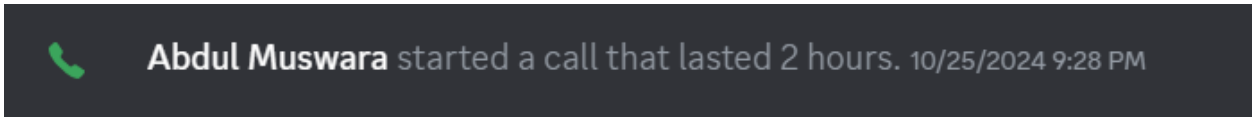


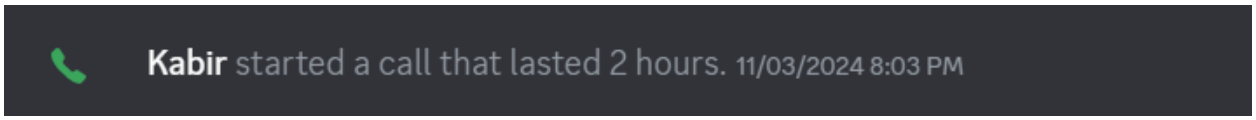



E-BIDDING
Home
Contact
Login/Signup
My Biddings
No biddings found.
My Listings
No listings found.


localhost:3000/contact
E-BIDDING
Home
Contact
Login/Signup
Our Team
100 x 100
John Doe
Developer
LinkedIn Profile
100 x 100
Jane Smith
Designer
LinkedIn Profile

6. Memos of group meetings and possible concerns of team work
    - Constant communication via Project Group Chat
    - Periodical live meetings in person and over voice call
    - Discussions concerning design and methods to complete the project





Concern: Since one of our group members had to drop out, the division of labor could be in disarray. We will have to cover for the parts he was planning to do and we have to do it efficiently in order to fulfill the requirements of the project. Not doing so may lead to issues with the final product.

7. Address of the git repo (github, gitlab, bitbucket, etc) of your team's work so far - put all materials including this report there

https://github.com/mohammadkabir2003/ebay-project