
Four Pixels

**SmartBid
Software Requirements Specification
For Web Application**

Version <2.0>

SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

Revision History

Date	Version	Description	Author
15/10/2024	1.0	Build the initial report, clarify the scope and purpose of the project, detail the specific and supplementary requirements, and identify the actors and use-case reports and models.	Sehr Abrar, Saanavi Goyal, Srewashi Mondal, Debasree Sen
12/11/2024	2.0	Expand on introduction, adding and editing use cases, add an E-R diagram, include pseudocode for main functionalities, show detailed GUI of the system, and including team meeting information and GitHub.	Sehr Abrar, Saanavi Goyal, Srewashi Mondal, Debasree Sen

SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

Table of Contents

1.	Introduction	5
1.1	Purpose	6
1.2	Collaboration Class Diagram	
2.	Use Case	7
2.1	Use-Case Model Survey	7
2.2	Assumptions and Dependencies	8
3.	ER Diagram	8
4.	Detailed Design	8
4.1	Use-Case Reports	8
4.2	Supplementary Requirements	22
4.3	System Screens	22
4.4	Pseudocode	27
5.	Supporting Information	33

SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

Software Requirements Specification

1. Introduction

1.1 Purpose

The purpose of the E-Bidding System Software is to develop a fully functional platform that facilitates secure and efficient bidding transactions among its users. The system will allow users to create and manage personal accounts, bid on items and services, conduct secure transactions, and post listings for sale or rent. The software will incorporate role-based permissions, offering distinct profiles for visitors (V), users (U), and super-users (S), each with specific privileges and responsibilities on the platform. On the platform, key system behaviors will include bidding processes, rating mechanisms, and comprehensive account management. Overall, the system will ensure a seamless user experience, promoting safe transactions and fostering trust between participants.

1.2 Scope

The e-bidding system includes three types of users - visitors (V), users (U), and super-users (S) to facilitate safe, online transactions among the different user types. The visitors will have the ability to browse the listings and become users, the users will be able to post listings, bid on existing items and services, and manage their transactions. Lastly, the super-users will oversee the user approval, manage transnational disputes, and make decisions based on user ratings. Moreover, the project will have features to ensure user authentication, secure monetary transactions, ratings, and personalized feed based on user activity. Lastly, there will be VIP users who will be promoted based on a predetermined criteria based on the account activity. All in all, the details above outline the system's functionality and behaviors using a use-case model, detailing interactions such as bidding, account management, and complaint handling. It will guide the design, development, and testing phases to ensure the system meets all functional and non-functional requirements.

1.3 Definitions, Acronyms, and Abbreviations

- V - Visitor: a user type who can browse through the listings on the mini-bidding platform, but must apply to be a user.
- U - User: a user type with the ability to list and bid products and services, and conduct transactions.
- S - Super-User: a user type with admin responsibilities to approve visitors to be users, manage ratings, and handle complaints for a seamless user experience.
- VIP: a user role given to users who fulfill certain requirements defined later, and have additional privileges than a regular user.

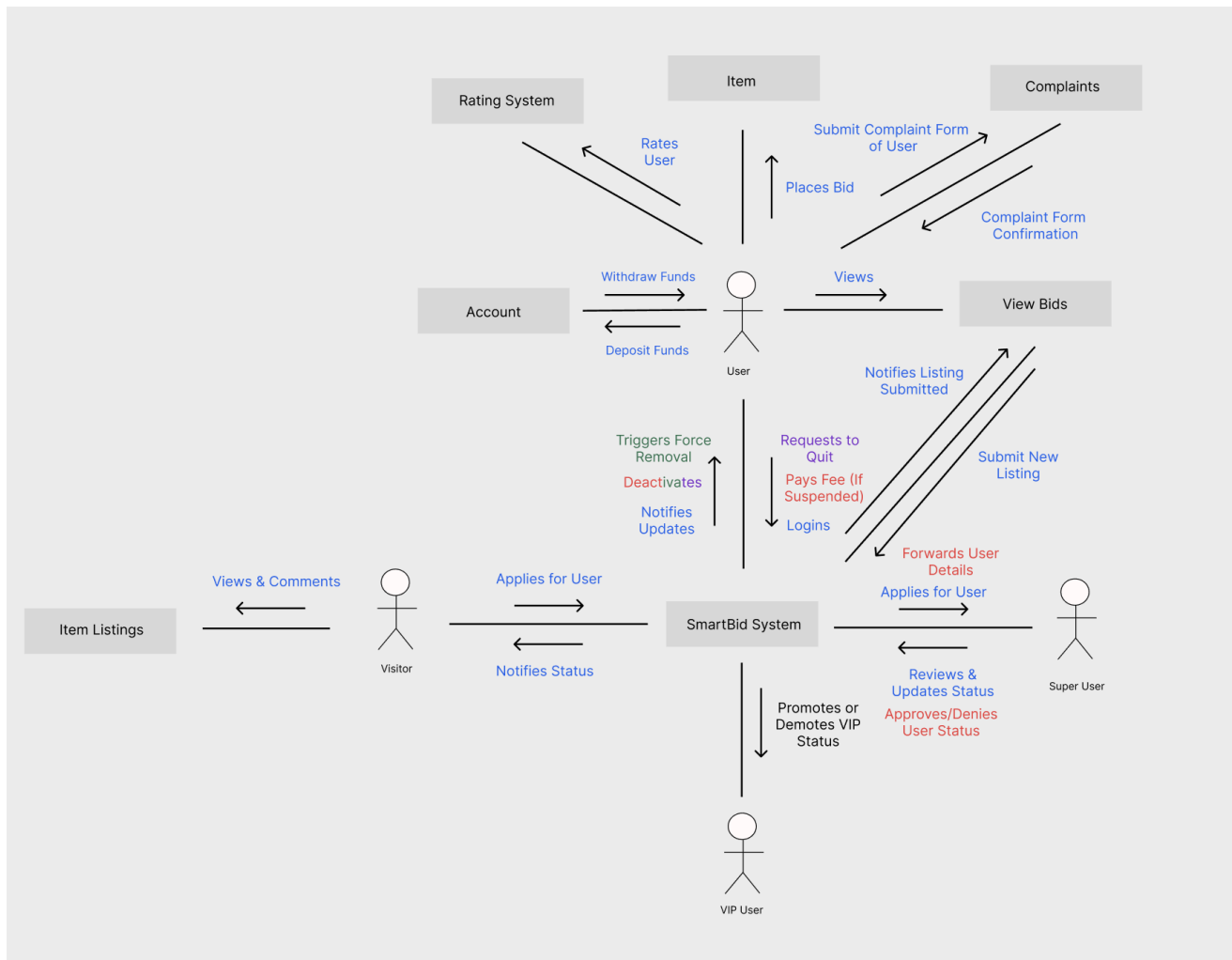
1.4 References

IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.

1.5 Overview

This Software Requirements Specification organizes four distinct sections to provide a comprehensive understanding of the e-bidding system, its requirements, and functionalities. These sections include an introduction to the project's purpose, scope, and overview, a detailed description of its use-case models and reports, the dependencies and assumptions, and specific and supplementary requirements. Each section focuses on key aspects such as design constraints and other essential details to aid in the successful development of the system. Overall, the document is designed to ensure that readers can easily navigate through the different sections and gain a clear understanding of the system's architecture and requirements.

1.2 Collaboration Class Diagram

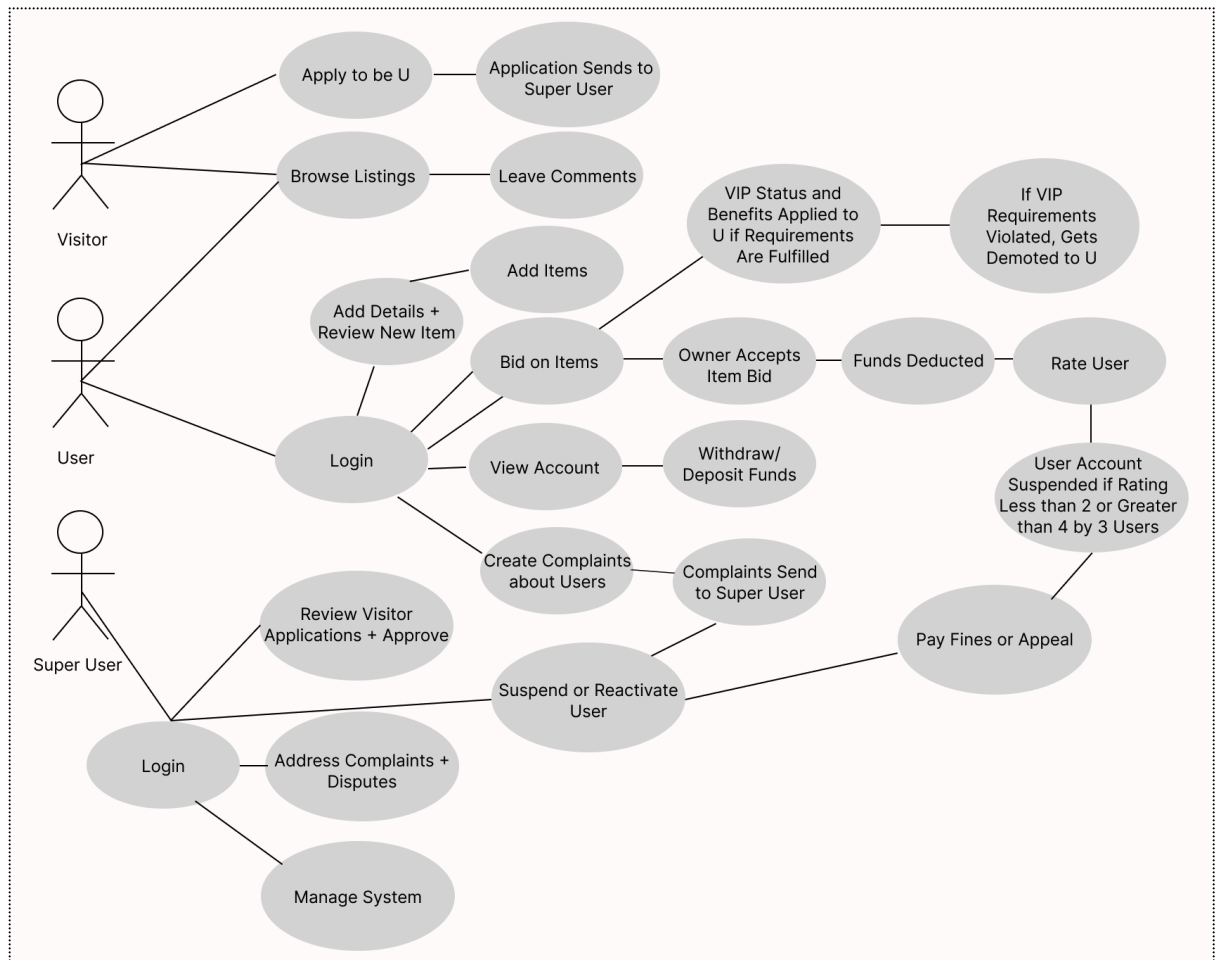


SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

2. Overall Description

2.1 Use-Case Model Survey

Use-Case Model:



Actors:

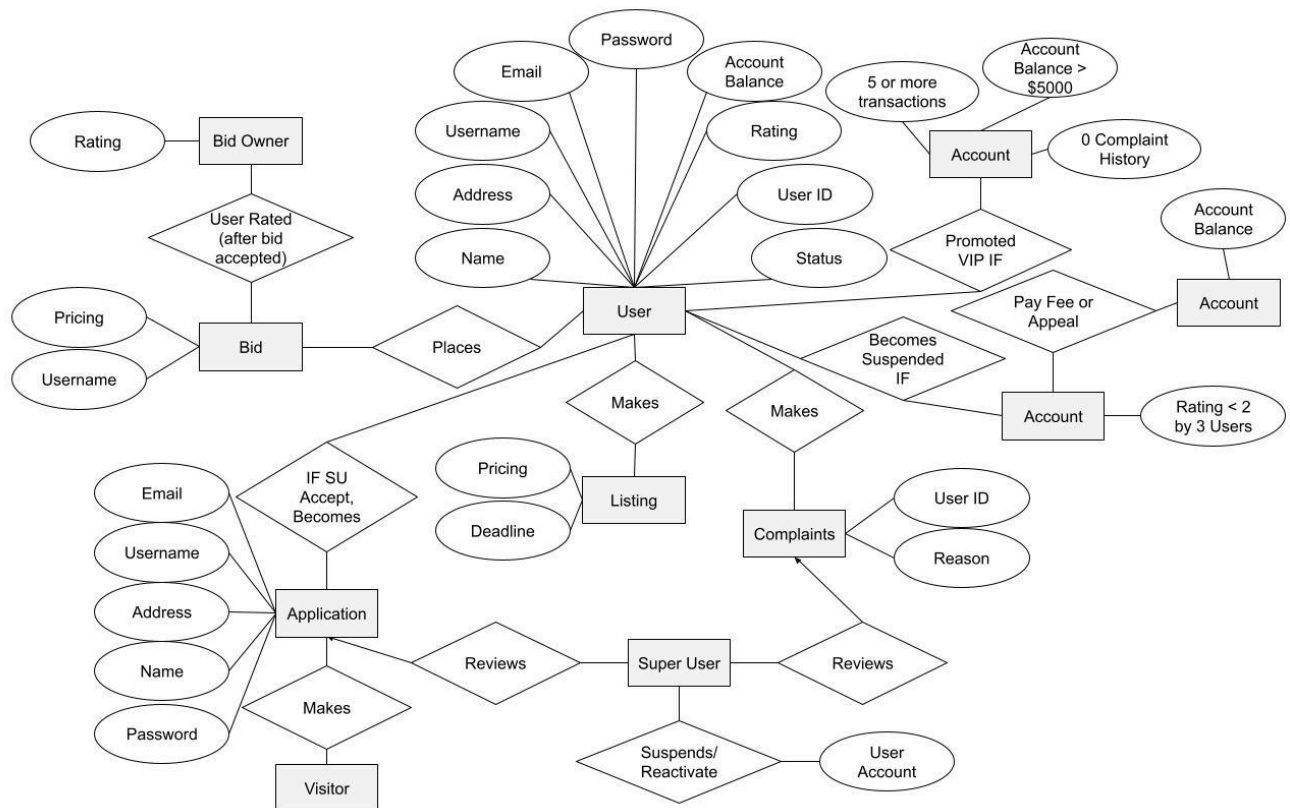
- Visitor (V)
 - can browse through listings of items and/or services
 - can leave comments
 - can apply to become User (U) once they verify that they are not a robot
 - cannot perform transactions nor bid on items or services
- User (U)
 - can manage account balance (deposit and withdraw permissions)
 - can list items and services for sale with needed descriptions (price ranges)
 - can bid on available items or services
 - can accept or decline bids for their items or services
 - can finalize transactions when the item owner accepts the bid
 - can anonymously rate other Users post-transaction
 - can submit complaints about other users to super-users if they want
 - is suspended if rating falls below 2 based off of at least three users
 - can pay fines/appeal to super-user to be reactivated if suspended

- Super-User (S)
 - can approve applications from visitors (V) to become users (U)
 - can address complaints and disputes
 - can suspend/reactivate users (U) from ratings or complaints
 - can manage both system and user behavior
- VIP
 - demoted to User if suspended 3 times or ratings are less than 2
 - have completed at least five transactions
 - have more than \$5,000 in their account
 - have a complaint-free transaction history
 - are shielded from suspension
 - earn a 10% discount on transactions

2.2 Assumptions and Dependencies

One assumption is that the system assumes users will act ethically and participate only in fair transactions. Any misconduct will be flagged by other users through the rating and complaint mechanisms. Another assumption is that visitors (V) who wish to become users (U) are expected to provide valid information during the registration process. One dependency is that the system relies on third-party payment providers to securely process financial transactions. Another dependency is an arithmetic equation/question that will be used to prevent bots during the visitor registration process.

3.0 ER Diagram

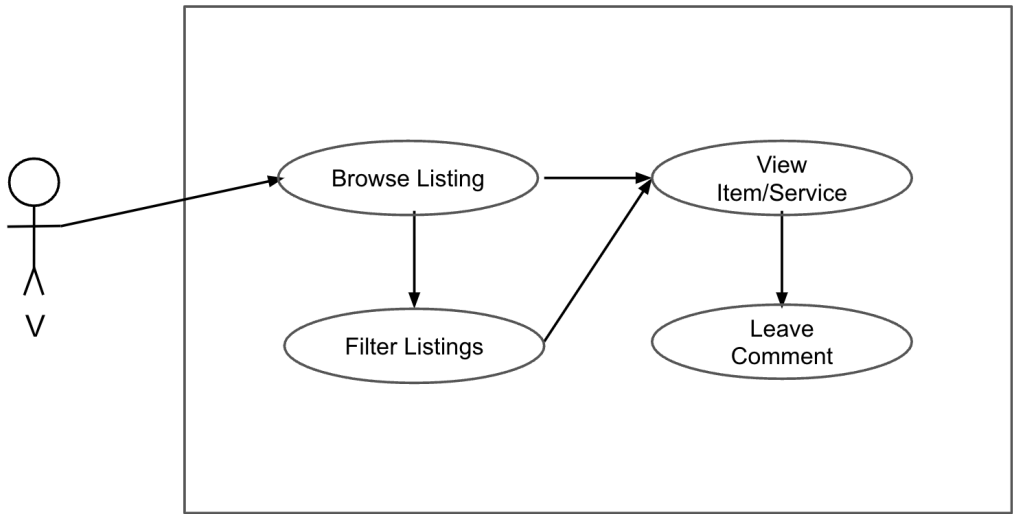


SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

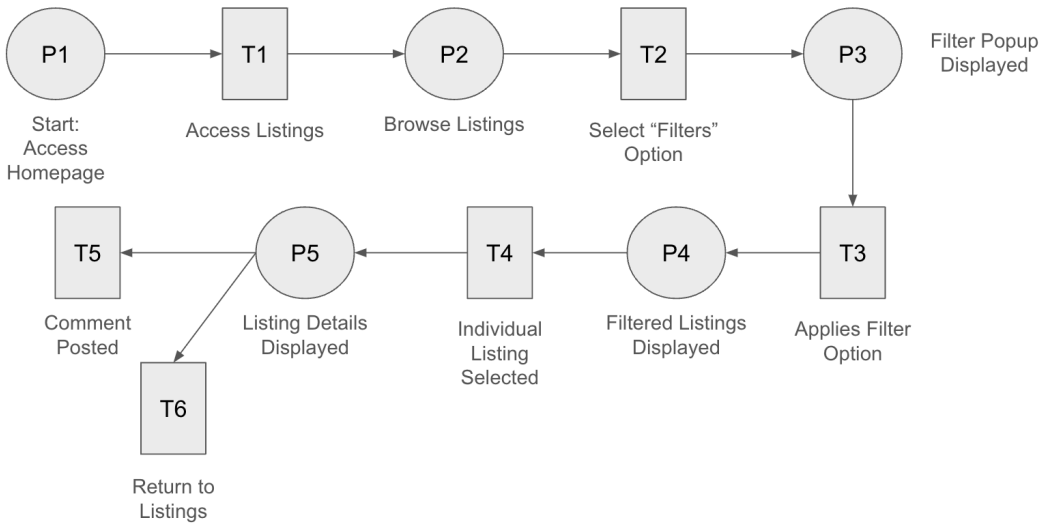
4. Specific Requirements

4.1 Use-Case Reports

Use Case 1: Visitor Browses Listings
Diagram



Petri-net Diagram



Collaboration Diagram



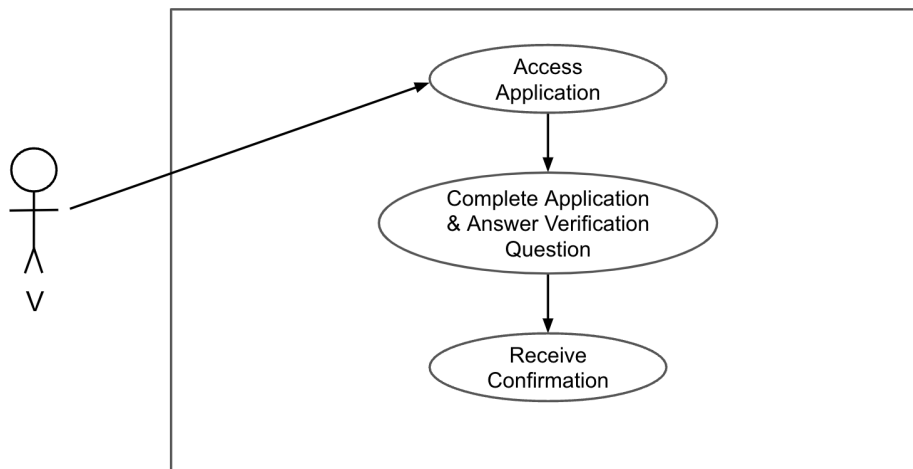
Brief Description & Precondition: A V navigates to the homepage, selects the “Browse Listings” option, and views a list of available items and services posted by U’s. The V can filter the listings, view detailed information about individual items, and leave comments. The precondition is that the V must have access to the internet and navigate to the system’s homepage.

Step-By-Step Description:

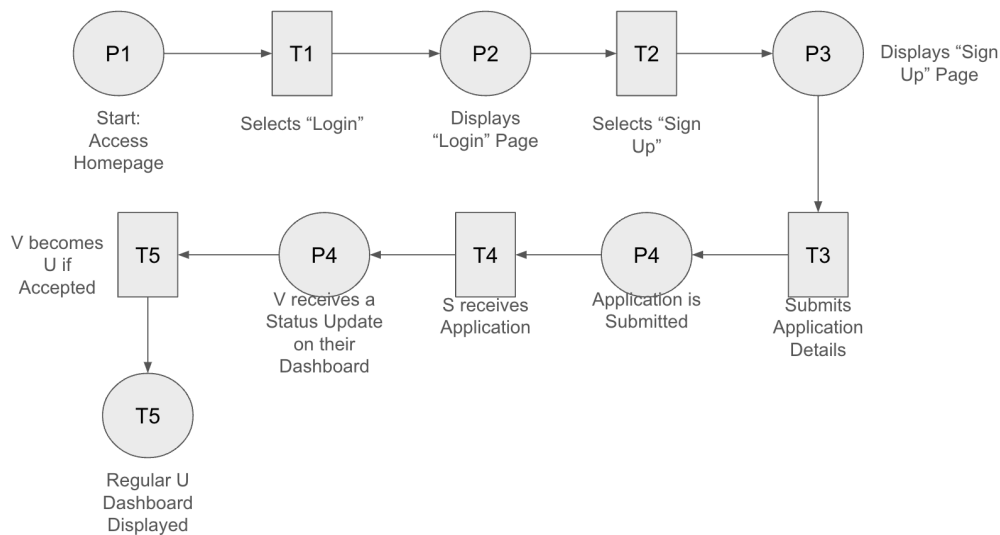
1. The V navigates to the homepage of the system.
 2. The V selects the “Browse Listings” option, which initiates the display of available items/services.
 3. The system retrieves and displays a list of available items/services posted by U.
 4. The V can apply filters (e.g., categories, price range, location) to narrow down the listings to their preferences.
 5. The V selects an individual listing to view its details, including information, pictures, price, and any other relevant details.
 6. The V has the option to leave a comment on the item/service.
 7. The V can return to the list of available items/services after viewing details.
- **Normal Use Case:** V successfully navigates to the homepage, selects "Browse Listings," and views a list of items/services. They apply filters, select a specific listing to view details, and have the option to leave a comment. V can then return to the main list of items.
 - **Exceptional Use Case:** If the system encounters issues (e.g., server error, internet connection loss), listings may fail to load, filters may not apply correctly, or comments may not be posted. V receives an error message or prompt to reload the page.

Use Case 2: Visitor Applies to Become a User

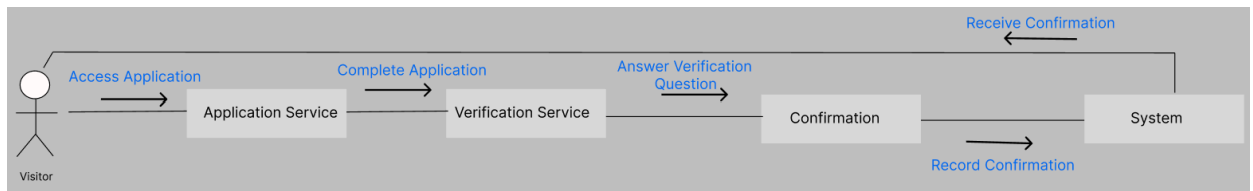
Use-Case Diagram



Petri-net Diagram



Collaboration Diagram:



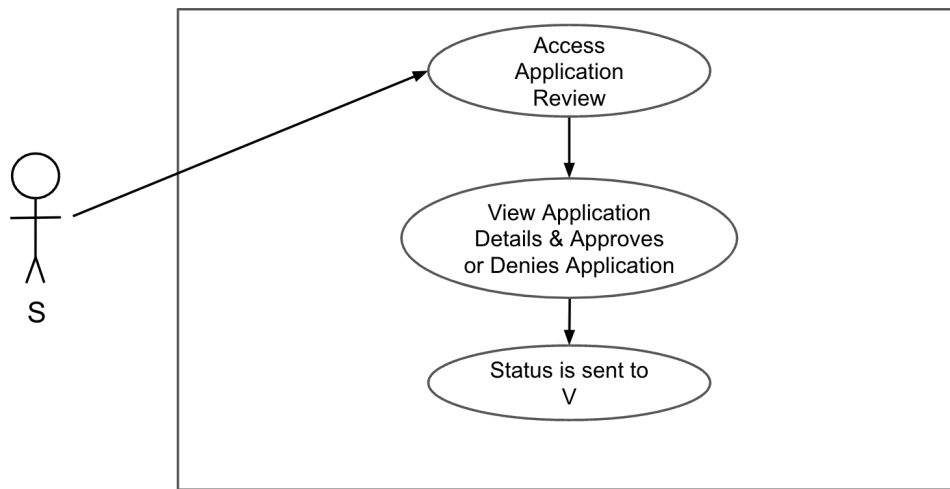
Brief Description & Precondition: A V can submit an application to become a U of the system. The application includes answering a random arithmetic question to verify the applicant is a human and not a robot. After submitting, the application is sent to a Super-User for review and approval or denial. The preconditions are the following: the V is not currently a registered U of the system, the V has access to the application form, and the V is able to answer the arithmetic question to prove that they're human.

Step-By-Step Description:

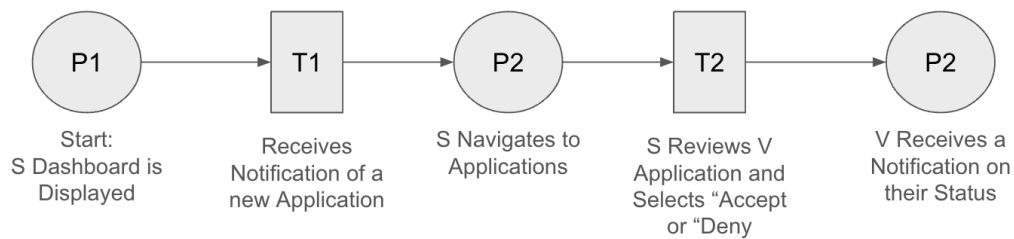
1. The V navigates to the application section of the website.
2. The V fills out the application form, providing necessary personal information (name, email), and then answers the random arithmetic question presented to confirm their human, before submitting.
3. Once the V submitted their application, the system displays a confirmation message that the application has been submitted successfully, notifying a Super-User that a new application is pending review.
 - **Normal Use Case:** V navigates to the application section, fills out the application form, provides necessary details, answers the arithmetic question, and submits the form. The system displays a confirmation message, indicating that the application has been sent to a Super-User for review.
 - **Exceptional Use Case:** If there are issues during submission (e.g., incorrect arithmetic answer, incomplete form, system error), the application is not submitted. The system notifies V of the specific error, prompting them to correct the information or answer the question again before resubmitting.

Use Case 3: Super-User Approves or Denies Application

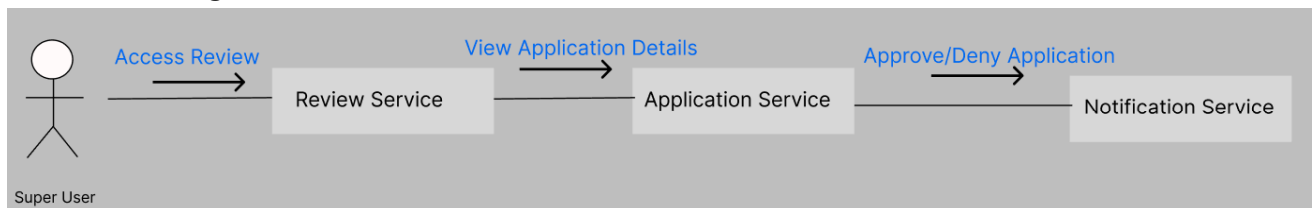
Use-Case Diagram



Petri-net Diagram



Collaboration Diagram



Brief Description & Precondition: A S reviews a V's application to become a U of the system. The S evaluates the application details and makes a decision to either approve or deny the application. This outcome is then communicated to the V. The preconditions are the following: The S has access to the admin interface of the system, there is a pending application from a V awaiting review, and the S is authenticated and authorized to make decisions on applications.

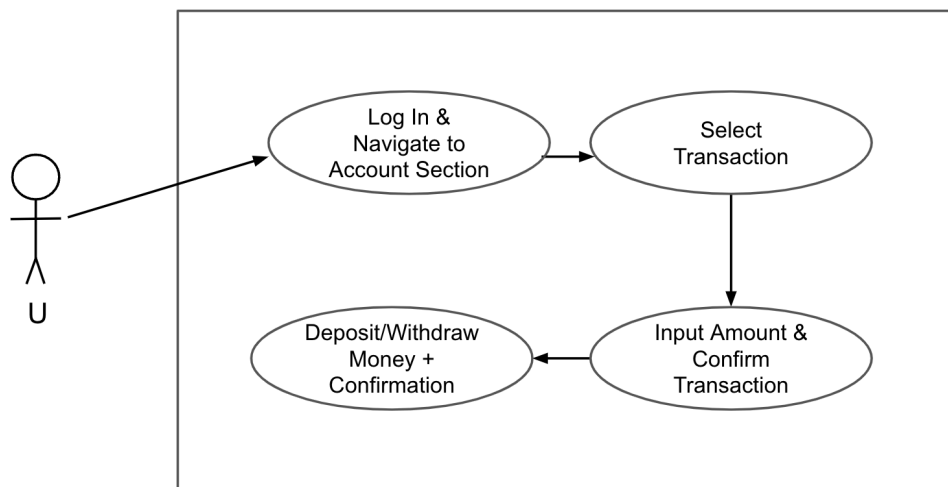
Step-By-Step Description:

1. The S receives a notification about a new application pending approval from a V.
2. The S navigates to the application review section of their admin interface.
3. The S selects the application to review, accessing the V's personal information and answer to the verification question.
4. The S assess the application based on the information provided and makes a decision:
 - a. Approve: If the application meets the criteria, the S approves the application.
 - b. Deny: If the application does not meet the criteria, the S denies the application with feedback.
5. Then the system sends a notification to the V regarding the status of their application.

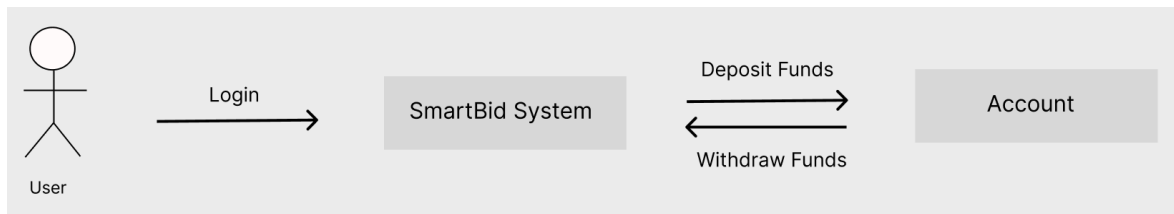
- **Normal Use Case:** S receives a notification for a pending application, navigates to the review section, and accesses the V's application details. S evaluates the application and either approves or denies it. The system then sends a notification to the V with the decision.
- **Exceptional Use Case:** If there is an issue (e.g., system error, missing application details, or loss of access to the admin interface), the S cannot complete the review. The system alerts the S of the issue, and the application status remains pending until the issue is resolved or the S retries the process.

Use Case 4: User Deposits/Withdraws Money

Use-Case Diagram



Collaboration Diagram



Brief Description & Precondition: A U deposits money into their account or withdraws money from it. The U can access their account balance, select the desired transaction type, input the amount, and confirm the transaction. The system updates the account balance accordingly and notifies the U of the successful transaction. The preconditions are as follows: the U must have an active account in the system, the U must be logged in to access their account, the U must have a valid payment method linked to their account for a deposit, the U must have sufficient funds in their account for a withdrawal.

Step-By-Step Description:

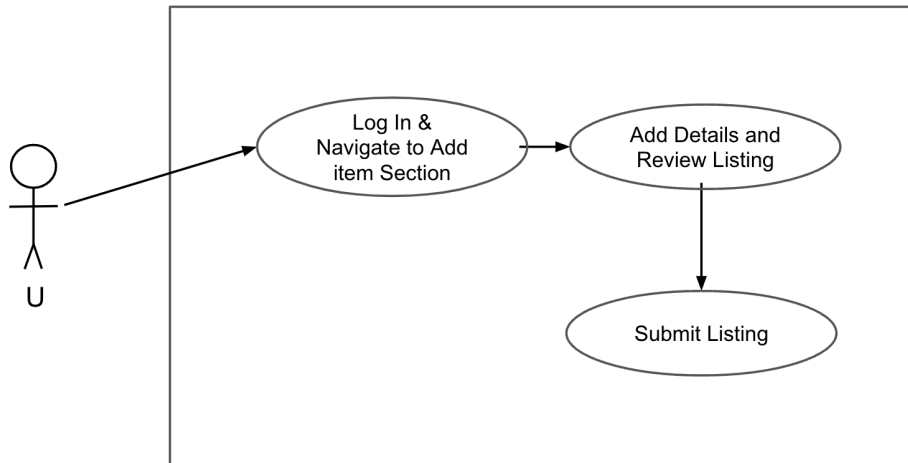
1. The U logs into their account and navigates to the account management section.
 2. The U chooses to either deposit or withdraw money and enters the amount they wish to deposit or withdraw.
 - a. The U confirms the deposit, and the system processes the transaction.
 - b. The U confirms the withdrawal, and the system checks for sufficient funds before processing.
 3. The system updates the U's account balance based on the transaction, and displays a message confirming the deposit or withdrawal with the updated balance.
- **Normal Use Case:** U logs in, navigates to the account management section, and selects either deposit or withdrawal. They enter the transaction amount and confirm. For deposits, the system processes the

transaction and updates the balance. For withdrawals, the system checks for sufficient funds, processes the transaction, updates the balance, and confirms success with the U.

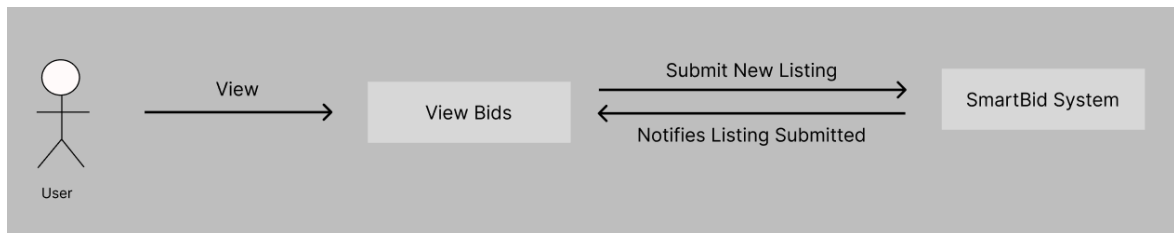
- **Exceptional Use Case:** If there is an issue (e.g., system error, missing application details, or loss of access to the admin interface), the S cannot complete the review. The system alerts the S of the issue, and the application status remains pending until the issue is resolved or the S retries the process.

Use Case 5: User Adds Item for Sale/Rent

Use-Case Diagram



Collaboration Diagram



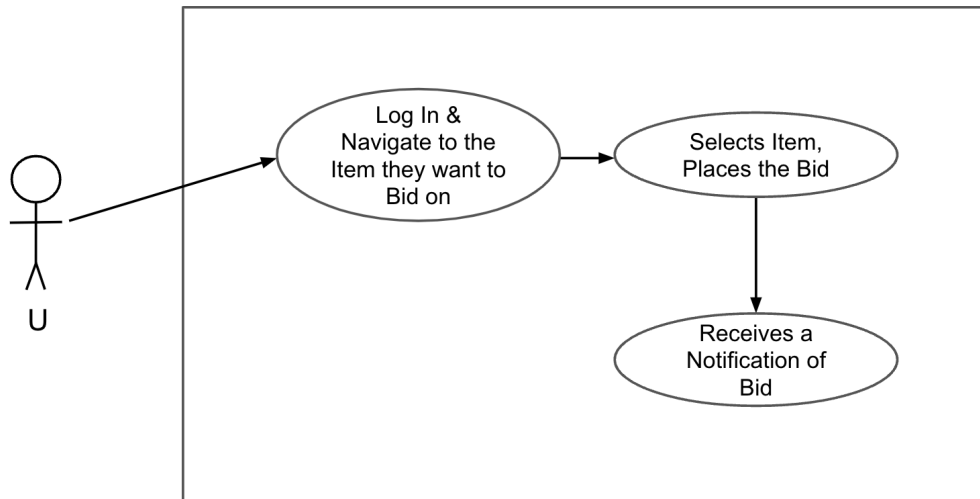
Brief Description & Precondition: The U can list an item or service for sale or rent on the platform by providing the necessary details, including a description, price, and any relevant conditions, and submits the listing for approval before it becomes visible to other U. The preconditions are as follows: the U must have an active account (and logged in), and the item or service listed must meet the platform-specific guidelines.

Step-By-Step Description:

1. The U logs into their account and navigates to the section for adding items/services for sale.
 2. They add the item or service and enter the following: the title, detailed description, price, accepted price range, any relevant images of the item/service, deadline, and the category. Then they review their listing before submitting.
 3. They receive a notification that their listing has been submitted.
- **Normal Use Case:** U logs in, navigates to the listing section, and enters details (title, description, price, images, deadline, and category). After reviewing, U submits the listing, and the system displays a notification confirming that the listing has been successfully submitted for approval.
 - **Exceptional Use Case:** If there is an issue (e.g., missing required information, violation of platform guidelines, or system error), the listing submission fails. The system alerts U of the specific issue, prompting them to correct details and resubmit.

Use Case 6: User Bids on Items

Use-Case Diagram



Collaboration Diagram



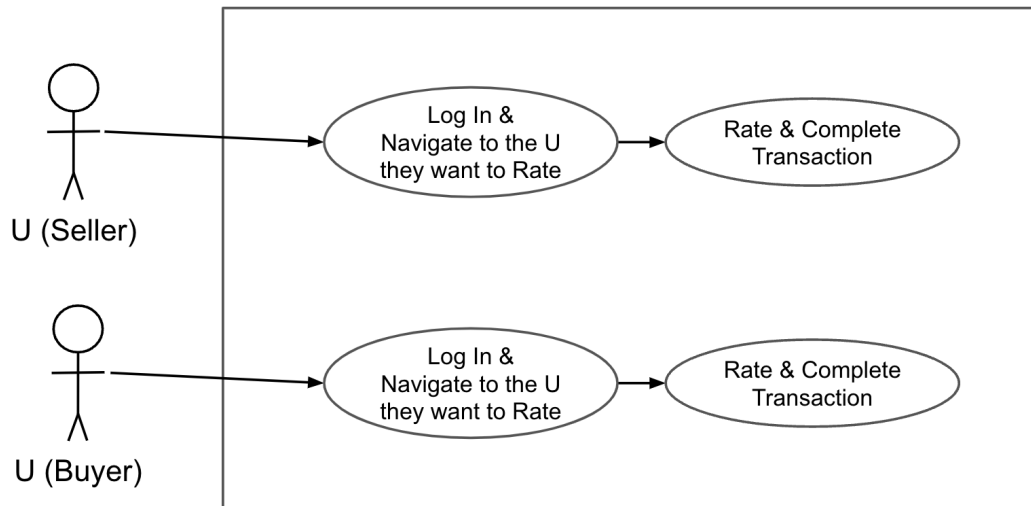
Brief Description & Precondition: A U places bids on available items within the marketplace and must have sufficient funds in their account to participate in bidding. The preconditions are the following: the U must have an active account (and be logged in), the item must be available for bidding, and the U must have sufficient funds in their account to place a bid.

Step-By-Step Description:

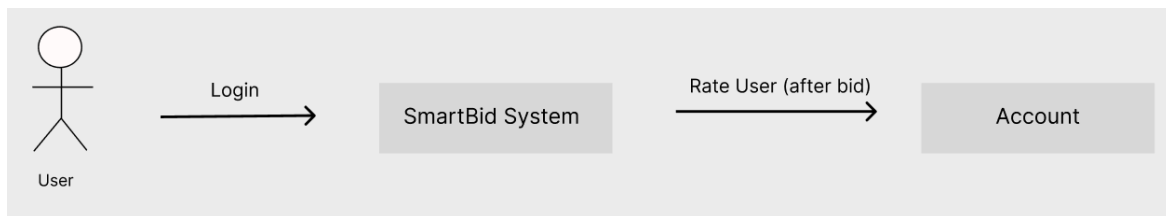
1. The U logs into their account and navigates to the list for bidding.
2. The U selects an item they wish to bid on,
3. The U enters their bid amount, confirms the bid, the system verifies they have sufficient funds, and the system updates the bid and notifies other U's on the bid amount.
4. The system notifies the U of a successful bid placement and updates the bidding status of the item.
 - **Normal Use Case:** U logs in, navigates to the bidding section, selects an item, enters their bid amount, and confirms the bid. The system verifies sufficient funds, updates the bid, notifies other users, and confirms the successful bid to U with an updated bidding status for the item.
 - **Exceptional Use Case:** If there's an issue (e.g., insufficient funds, item no longer available for bidding, or system error), the bid is not placed. The system alerts U of the issue (e.g., "Insufficient funds" or "Item unavailable"), allowing U to adjust or retry as necessary.

Use Case 7: Transaction Ratings

Use-Case Diagram



Collaboration Diagram



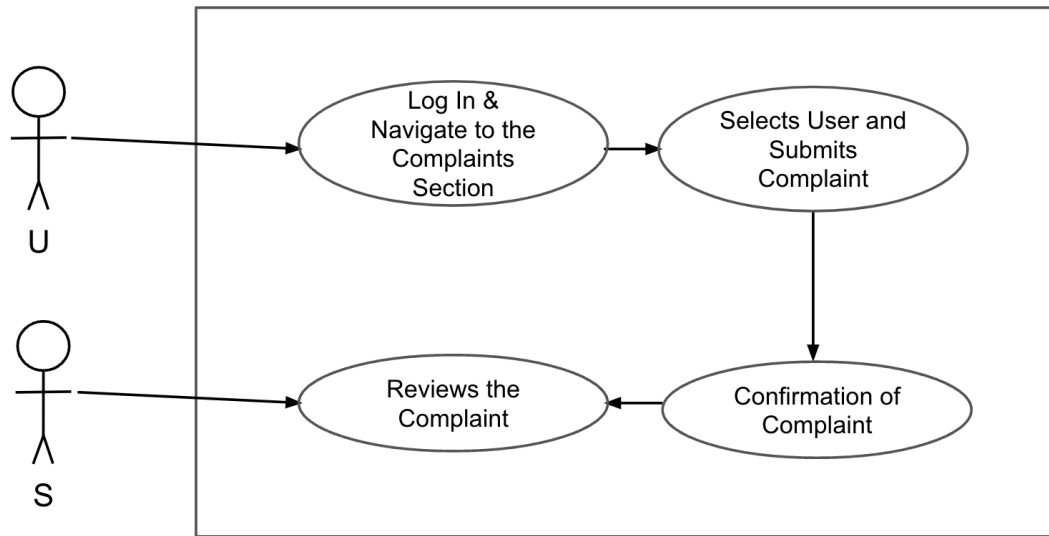
Brief Description & Precondition: The U's can rate their transaction experience after completing a purchase or a rental. Ratings are submitted anonymously to promote unbiased feedback. The preconditions are that the U must be active and logged in and the U must be completed for the U to rate it.

Step-By-Step Description:

1. The U logs into their account and completes a transaction.
2. The U who bought the item/service has the option to rate the seller and vice versa. They both rate their experience based on their experience (1 to 5 stars). Both ratings are submitted anonymously to the system.
3. The system then confirms that the ratings have been recorded.
 - **Normal Use Case:** After completing a transaction, U logs in and accesses the rating option. They submit a rating (1 to 5 stars) for the other party anonymously, and the system confirms that the rating has been successfully recorded.
 - **Exceptional Use Case:** If there's an issue (e.g., transaction not marked as completed, system error, or attempt to submit multiple ratings for the same transaction), the rating cannot be recorded. The system notifies U of the specific issue (e.g., "Transaction not completed" or "Rating already submitted") to resolve before retrying.

Use Case 8: User Complains about Another User

Use-CaseDiagram



Collaboration Diagram



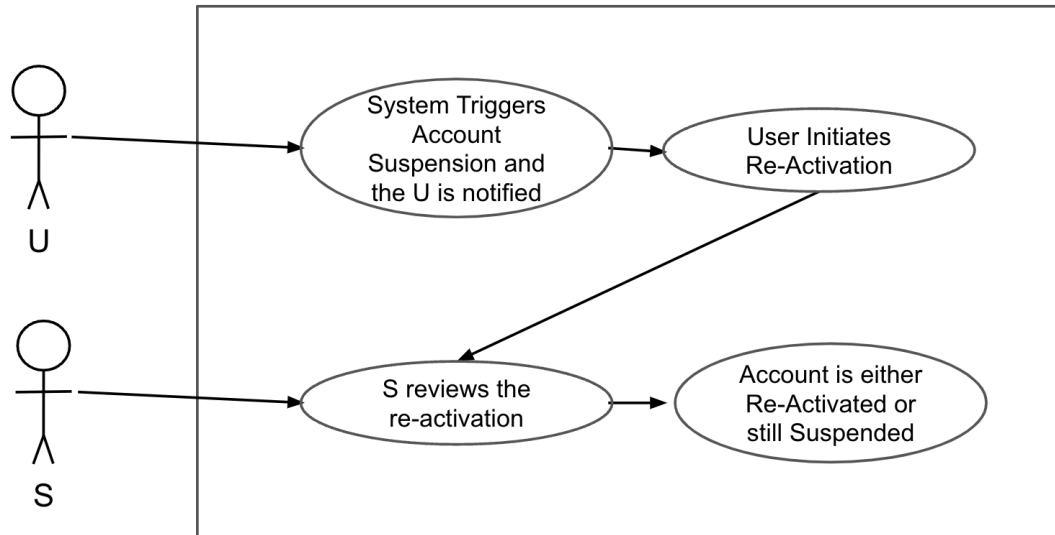
Brief Description & Precondition: A U can submit a complaint against another U regarding a transaction or behavior. The complaint is then reviewed by a S (system administrator). The preconditions are the following: the U must have an active account and be logged in, the U must have completed a transaction with the U they wish to complain about, the complaint must be on a specific criteria.

Step-By-Step Description:

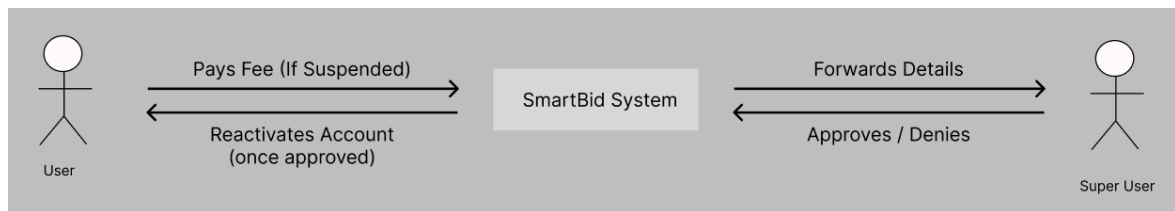
1. The U logs into their account and navigates to the complaint list of their recent transactions.
2. The U submits a complaint form detailing their issue.
3. The system sends a receipt of the complaint and notifies it will be reviewed.
4. The S reviews the complaint and takes appropriate action based on the findings.
 - **Normal Use Case:** U logs in, navigates to the complaint section, and submits a complaint form detailing the issue with a recent transaction. The system confirms receipt of the complaint and notifies U that it will be reviewed. S reviews the complaint and takes appropriate action based on the findings.
 - **Exceptional Use Case:** If there is an issue (e.g., incomplete complaint form, transaction not eligible for a complaint, or system error), the complaint cannot be submitted. The system notifies U of the specific issue (e.g., “Transaction not completed” or “Complaint criteria not met”) and prompts them to resolve the issue before resubmitting.

Use Case 9: Suspension and Re-Activation

Use-Case Diagram



Collaboration Diagram



Brief Description & Precondition: The U is suspended from the system due to low ratings or violation of system rules. This will also outline the steps for re-activating their account after the suspension period or payment of fines. The preconditions are the following: the U must have been flagged for suspension due to low ratings (or too high ratings) or multiple complaints, the U must have an active account prior to suspension, and if the suspension is due to ratings, the U's average rating must be below the specified threshold.

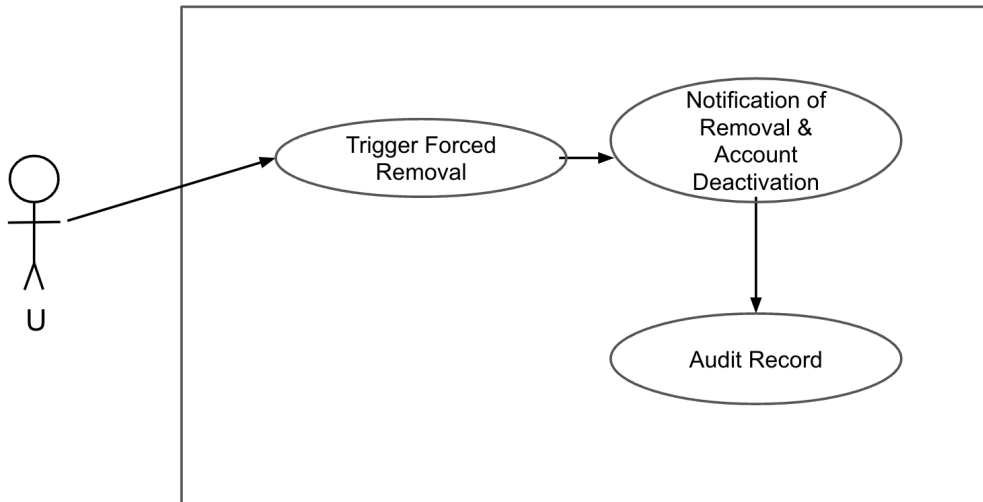
Step-By-Step Description:

1. The system checks the ratings after a transaction, if the average rating is below 2 or above 4 (evaluated by at least three U's), the U is flagged for suspension.
 2. The U is notified of their suspension status via notification which would include why they're suspended and how to initiate the reactivation process (via \$50 fine or S). When a U's account is suspended, they lose access to functionalities, including bidding or listing items. A record of their suspension is also created.
 3. An S reviews the suspension case, checking for any additional information or complaints, and they decide either to approve or deny the reactivation request.
 4. If approved, the U's account is reactivated, restoring full access and the U is notified. If denied, the U is informed of the decision and any further actions required.
- **Normal Use Case:** The system monitors the U's ratings after a transaction. If the average rating falls below 2 or exceeds 4 (evaluated by at least three users), the U is flagged for suspension. The U is notified of their suspension and informed of the reactivation process (e.g., payment of a fine or review by a system administrator). The suspension restricts access to functionalities like bidding or listing. S reviews the suspension case, approves or denies reactivation based on additional information, and notifies the U accordingly. If reactivated, the U gains full access again.
 - **Exceptional Use Case:** If there's an issue (e.g., failure to meet the reactivation requirements, incomplete fine payment, or error in the review process), the account reactivation may be denied. The

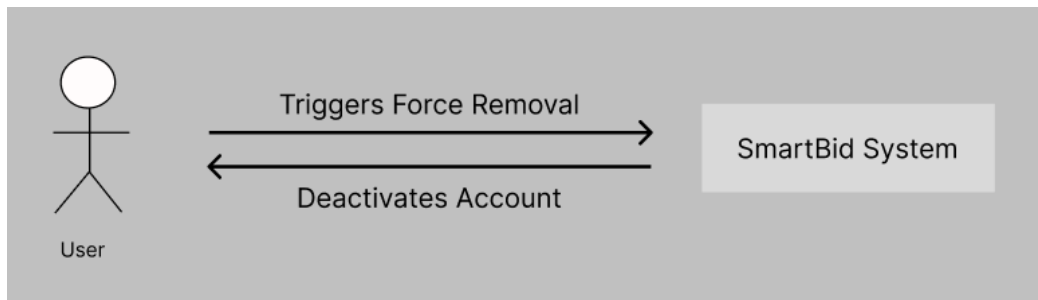
system notifies the U of the specific issue (e.g., “Fine not paid,” “Insufficient grounds for reactivation”) and outlines steps for further actions, such as paying fines or correcting the issue.

Use Case 10: Forced Removal of User

Use-Case Diagram



Collaboration Diagram



Brief Description & Precondition: The U is permanently removed from the system after repeated suspensions, The U’s access to the platform is revoked, and their data may be handled according to privacy policies. The preconditions are the following: the U must have been suspended three times for low ratings or violations of the system rules, the U must have received notifications regarding their suspensions and the consequences, the U’s account must be currently active prior to forced removal.

Step-By-Step Description:

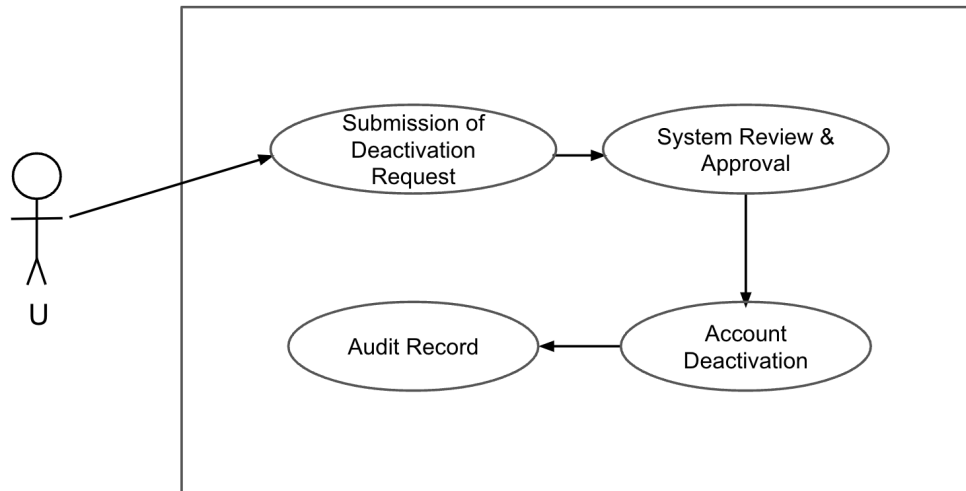
1. The system identified U who have been suspended three times, and a record of these suspensions is verified by the system.
2. The U receives a notification informing them of the forced removal of the platform, this notification would list the reasons for removal and any information regarding their account status.
3. The system then deactivates the U’s account, revoking all access and functionalities.
4. An audit record of the forced removal is created in the system.
 - **Normal Use Case:** The system identifies U’s who have been suspended three times for low ratings or violations of system rules. A record of these suspensions is verified. The U is notified of their forced removal, which includes the reasons for removal and account status information. The system

deactivates the U's account, revoking access to all functionalities. An audit record of the forced removal is created for system reference.

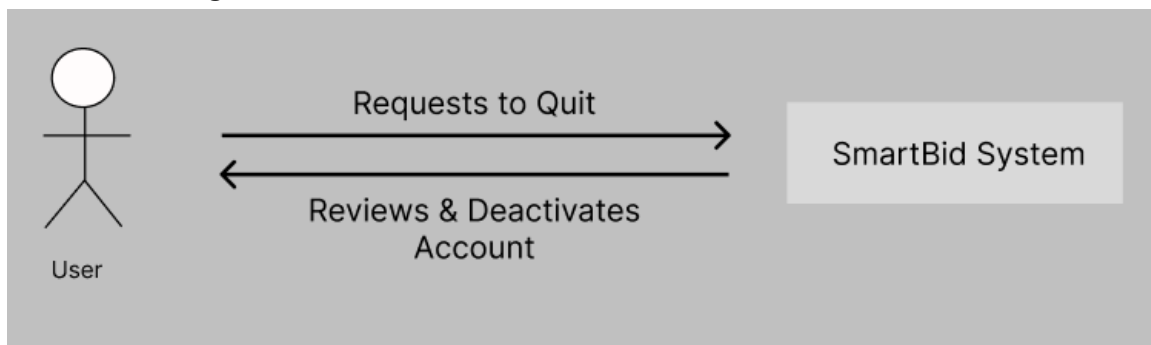
- **Exceptional Use Case:** If there's an issue (e.g., system error, failure to properly track suspensions, or data inconsistency), the forced removal cannot be completed as expected. The system notifies the U of the issue (e.g., "Suspension record error," "System issue with account removal") and provides guidance for further actions, such as contacting support.

Use Case 11: User Requests to Quit the System

Use-Case Diagram



Collaboration Diagram



Brief Description & Precondition: The U voluntarily requests to leave the online marketplace system, in which their account will be deactivated. The preconditions are the following: the U must have an active account, the U must not be under suspension or forced removal, and the U must initiate the request through account settings.

Step-By-Step Description:

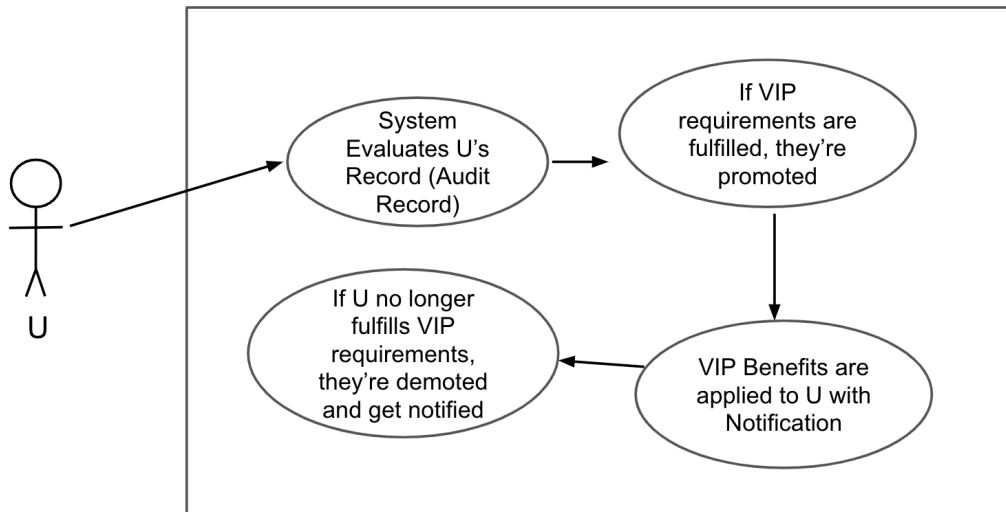
1. The U navigates to their account settings and selects the option to request to quit the system. A confirmation is followed shortly.
 2. The U submits the request to quit, which is logged in the system for processing.
 3. The system reviews the request for compliance with policies. If the U has any ongoing transactions, they will be notified to resolve these before proceeding.
 4. If it's approved, the U receives a notification that their account is deactivated.
 5. Then there is an audit recording this action.
- **Normal Use Case:** U navigates to their account settings and selects the option to voluntarily leave the system. A confirmation prompt appears, and the U submits their request to quit. The system logs the request and reviews it for compliance with policies, including checking for any ongoing transactions. If

the U has unresolved transactions, they are notified to address them. Once approved, the U receives a notification confirming the deactivation of their account. The system creates an audit record of the deactivation action.

- **Exceptional Use Case:** If there is an issue (e.g., ongoing transactions, system error, or failure to meet compliance requirements), the account deactivation request cannot be processed immediately. The system notifies the U of the issue (e.g., “Ongoing transactions must be resolved,” “Account deactivation failed due to system error”) and provides guidance for resolving the issue before the request can be resubmitted.

User Case 12: VIP Status Achieved

Use-Case Diagram



Collaboration Diagram



Brief Description & Precondition: The U could achieve VIP status within the online marketplace system, receiving special privileges and benefits as a reward for their loyalty and activity. The preconditions are as follows: the U must have an active account in the system, the U must achieve the criteria for VIP status, which includes having a balance of more than \$5,000, conducting more than 5 transactions, and not having any complaints against them.

Step-By-Step Description:

1. The system continuously monitors U accounts for eligibility based on predefined criteria (balance, transaction count, and complaint history).

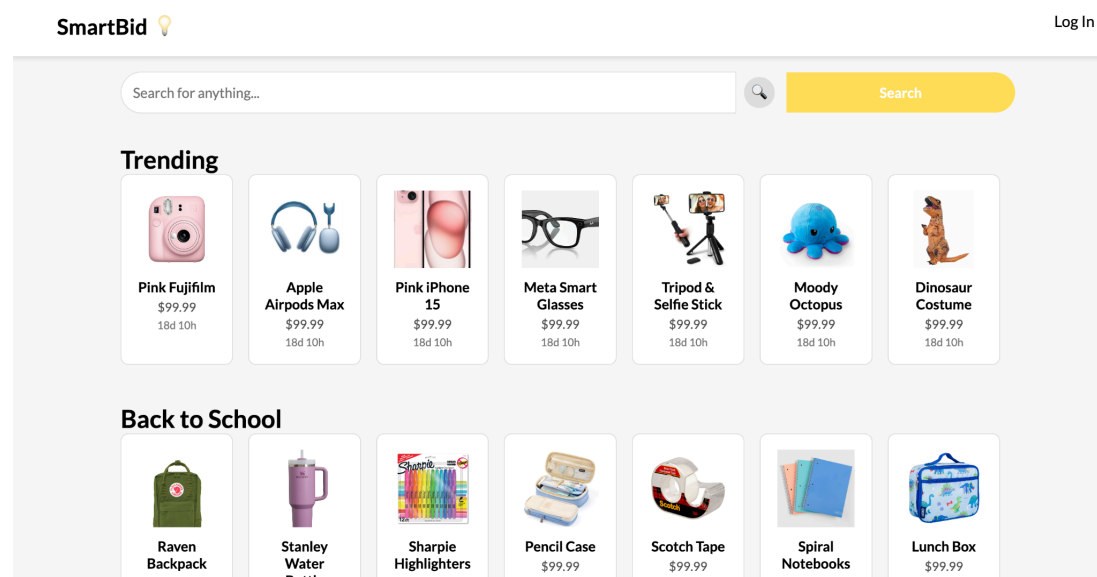
2. Upon completion of the VIP requirements, the system updates the User's account status to VIP and assigns the necessary privileges such as the exclusive discount (10% off).
3. The U receives a notification of their status update.
4. The system continues to monitor their activity, if the U fails to meet any two of the VIP criteria, they would be demoted back to a regular U.
5. An audit record is kept of their activity status.
 - **Normal Use Case:** The system continuously monitors U's accounts for eligibility based on predefined criteria (balance, transaction count, and complaint history). When a U meets the VIP requirements, the system updates their account status to VIP and grants special privileges, such as an exclusive 10% discount. The U is notified of the status update. The system continues to monitor the U's activity, and if they fail to meet at least two of the VIP criteria, their status is reverted to regular user. An audit record of their VIP status and activity is maintained.
 - **Exceptional Use Case:** If there is an issue with monitoring criteria (e.g., system error, failure to track balance or transactions), the VIP status update cannot be processed correctly. The system notifies the U of the issue (e.g., "Error in VIP status update" or "Transaction history not fully updated") and provides guidance for the U to resolve any discrepancies, or the system may request that they contact support for assistance.

4.2 Supplementary Requirements

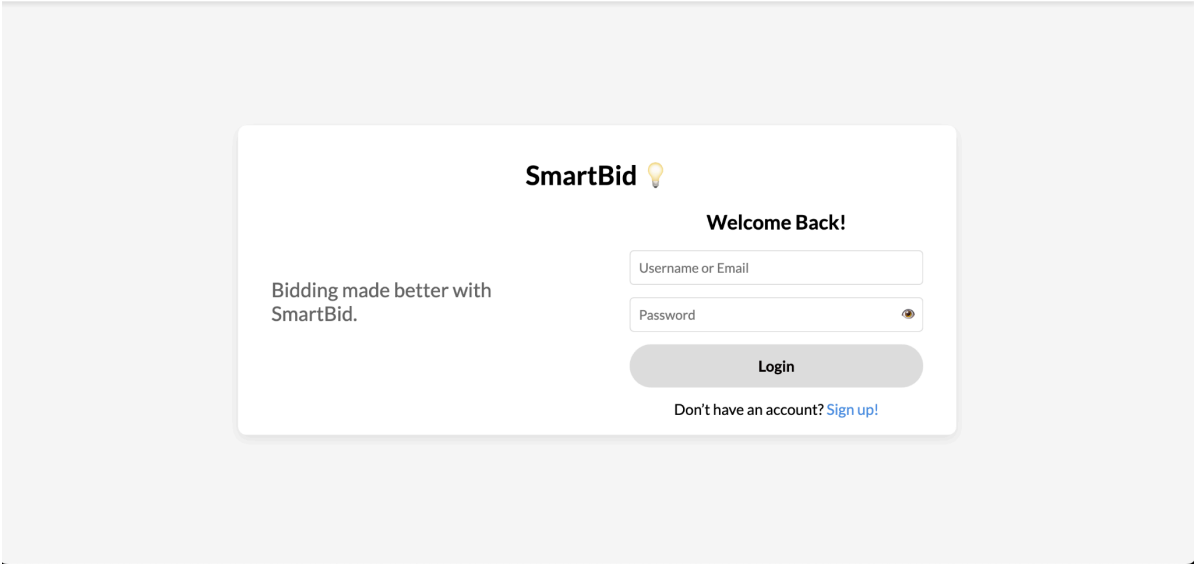
1. **Performance:** The system should deliver secure and efficient performance, handling key actions like transactions, browsing, and applications seamlessly.
2. **Security:** User data and sensitive actions should be encrypted to ensure privacy and security.
3. **User Interface:** The interface must be intuitive, providing easy navigation for common tasks such as browsing listings, placing bids, and managing accounts.
4. **Device Compatibility:** The system will be optimized for specific laptop devices, ensuring smooth usability, until future design changes are implemented.
5. **Reliability & Scalability:** The system should offer high reliability and be able to scale for future updates, allowing the addition of new features without disrupting existing functionalities.
6. **Error Handling:** Error handling should be clear and guide users toward resolving issues effectively.

4.3 System Screens

Homepage



Login



Listing Item





iPhone 15 Pro Max New

US \$99.99

Closes: 18d 10h (MM/DD/YYYY)

Category: Category Name

Seller: John Doe - 4.2 (10 Ratings)

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris.

[Start Bidding](#)

Leave a comment as a visitor

Cancel

Comment

Srewashi Mondal - 1 day ago

This product is nice.

▲ 10 ▼ 0

My Dashboard

My Account

View and update your personal information and account balance.

View Bids

View and update your current and past bidding transactions.

Requests

Manage your account status or report a user.

VIP Lounge

Learn what it means to be a VIP with SmartBid!

Notifications

New bid placed!

Account updated successfully.

My Account

Account Information

First Name:

Last Name:

Email:

Username:

Shipping Address:

Password:

Edit

Account Balance

Current Balance: \$0.00

Deposit

Withdraw

View Bids

Search for bid...

Search

Create Listing


Completed

In-Progress

Selling

Buying

Selling



Pink Fujifilm


US \$99.99

John Doe - 4.2 (10 Ratings)

In Progress

18d 10h

Buying



Apple AirPods Max

US \$99.99

Srewashi Mondal - 4.3 (12 Ratings)

In Progress

18d 10h

Requests (with Functionality)

Requests

Manage Account Status

Manage your status or pay a fee.

Account Status: Suspended

Deactivate Account

Current Fee: \$50.00

Pay Fee

Report User

File a complaint on another user.

File New Complaint

SmartBid

You

Requests

Manage Account Status

Manage your status or pay a fee.

Account Status: Suspended

Deactivate Account

Current Fee: \$50.00

Pay Fee

Report User

File a complaint on another user.

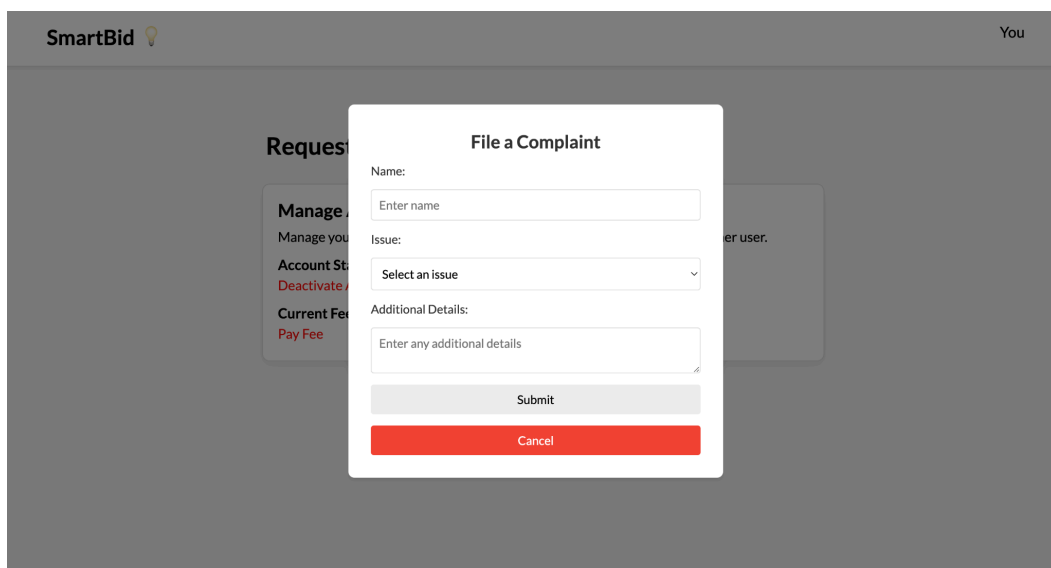
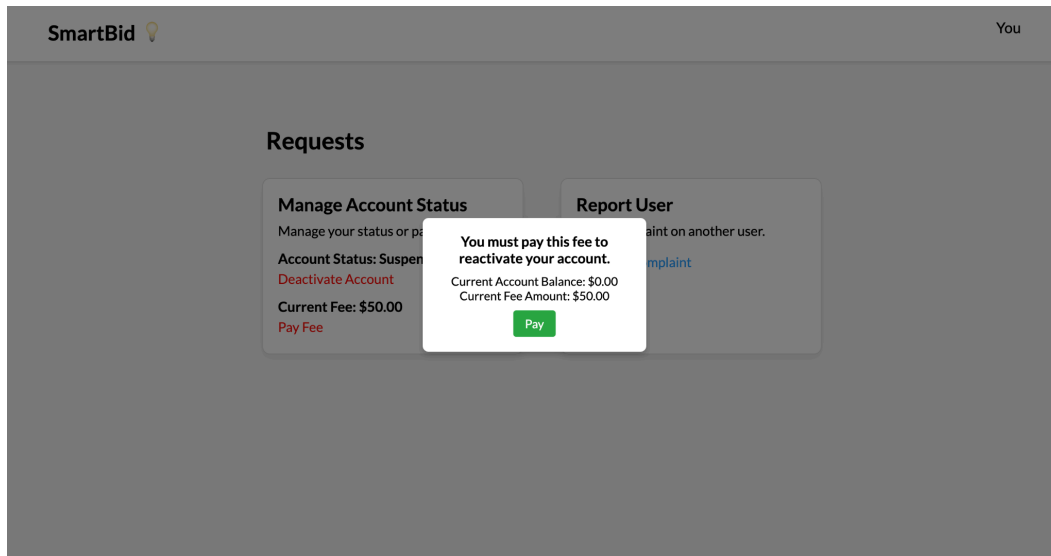
File New Complaint

Are you sure you'd like to deactivate your account?

You'll no longer have access to your account or it's data.

Yes

No



VIP Lounge

VIP Lounge

Congratulations! Learn more about what it means to be a VIP on SmartBid below!

VIP Perks

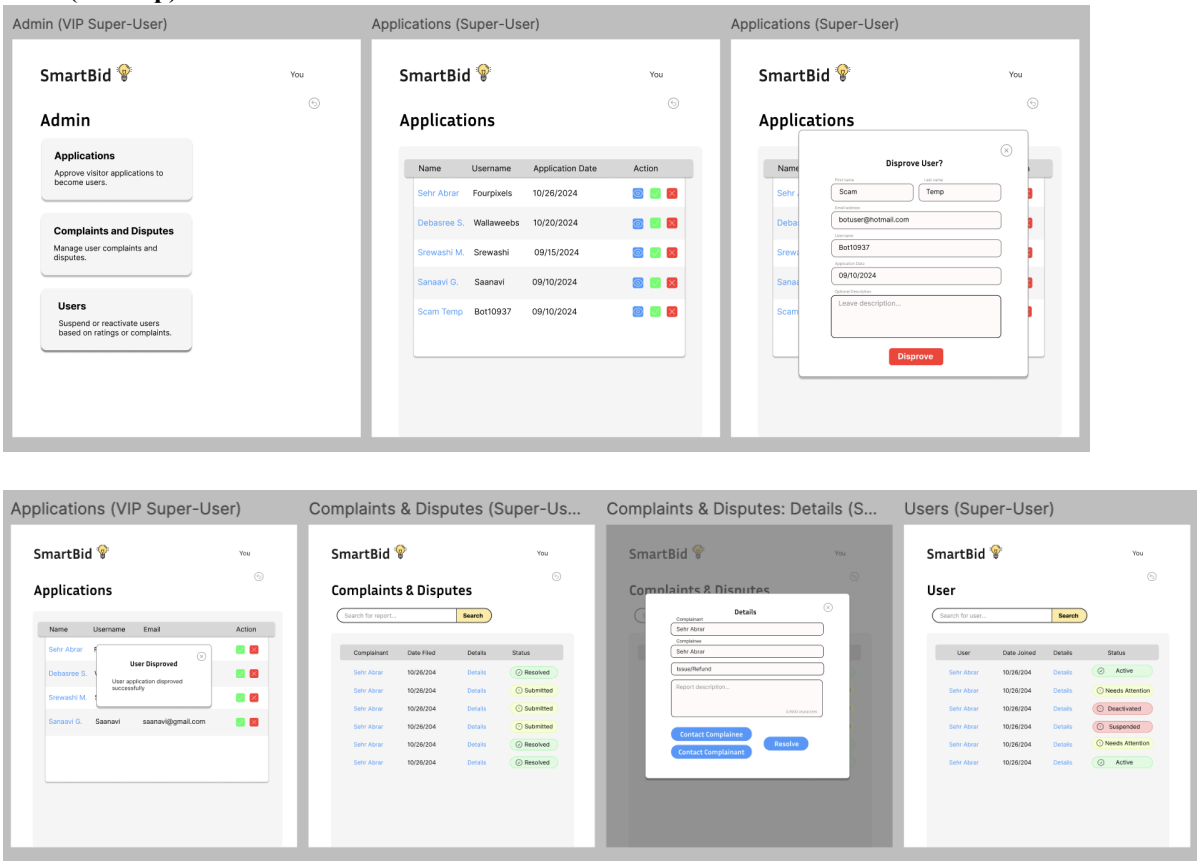
- 10% discount on every future transaction as VIP member.
- You'll now have access to items sold by VIPs for VIPs.
- Witness live bidding as a VIP.
- Your profile will now feature an epic VIP badge.

VIP Maintenance

- Maintain a balance of at least \$5,000 in your account.
- You must conduct more than 5 transactions.
- No users have filed a complaint against you.

If you violate any two of the above conditions, say bye to your VIP status!

Super User (mockup)



4.4 Pseudocode and Functionality

Use Case 1: Visitor Browses Listings

- Input:
- VisitorID (int value): a unique identifier number each visitor will have
- Output:
- AvailListing (array objects): it will output an array of all available listings which can be viewed by a visitor

Pseudocode for Functionality:

1. Create a boolean function isVisitor which will check if the visitorID is authorized to view the listings.
 - a. If Yes: All available, current and previous, listings will be retrieved from the database for display. The listings will also be returned as the output array.
 - b. If Not: Access to view the listings will be denied and the visitor will not be able to view any listings on the bidding website.

Use Case 2: Visitor Applies to Become a User

Input:

- VisitorID (int value): a unique identifier number each visitor will have
- ApplicationData (object): it will have all necessary data from the visitor, including their name, email, and any other information which might have been collected.
- securityAnswer (int): this will be an answer to a random arithmetic question which will confirm the status of a visitor. It will mostly check if the visitor is not a robot.

Output:

- ApplicationStatus (string): this string value will return if the visitor is eligible to apply and become a visitor, or not.

Pseudocode for Functionality:

1. The first step would be to check if the user is a visitor or not.
 - a. If not: access will be denied, and the string value will be outputted appropriately.
 - b. If yes: Two variables, securityQuestion and correctAnswer will be set to a predefined generated Security Question variable which will ask a random arithmetic question.
 - i. If the security answer matches the correct answers: a new applicationID will be created with the visitorID and their applicationData collected from the input. The application status will also be submitted, and will reflect appropriately.
 - ii. If not: The application status will be failed due to the incorrect answers, and the status will reflect accordingly.
2. Lastly, the applicationStatus will be returned.

Use Case 3: Super-User Approves or Denies Application

Input:

- superUserID (int): a unique identifier number each super user will have
- applicationID (int): this will be a unique ID number, previously generated in use-case 2, which will be used to represent each application uniquely
- status (string): this value will return if the application is approved or denied by the super user.

Output:

- ApplicationStatus (string): this string value will return whether the application has been approved, or rejected by the super user.

Pseudocode for Functionality:

1. Before approving or denying the application, a boolean function will need to confirm if the superUserID corresponds to the superuser access.
 - a. If not: access will be denied, and the applicationStatus will update appropriately.
 - b. Then, the application is retrieved using the applicationID and checked if the application has not been previously reviewed already.
 - c. If the application is not reviewed, and the user is a super user: a new user account will be created using the application, and the application status will be approved.
 - i. Similarly, the decisions can be denied, or if no proper output is received, it can be invalid.
2. Lastly, the application status will be appropriately outputted.

Use Case 4: User Deposits/Withdraws Money

Input:

- UserID (int): represents the unique ID for all approved users
- Amount (float): represents the transaction amount from the account.
 - Condition: A positive integer will indicate deposit, and a negative integral will be indicative of withdrawal.

Output:

- transactionStatus (string): will indicate if the transaction was successful or not.

Pseudocode for Functionality:

1. Use the UserID input to check to confirm that the user is an approved user.
 - a. If not: Access will be denied, and the output will show accordingly.
 - b. Then, the balance of the user is retrieved using their userID. We assume that the data like their transactional balances is already saved within the database for userIDs.

- c. If yes: the following can be done to deposit or withdraw money for bidding
 - i. If the amount is greater than 0, the balance is updated, and added to the amount. The transaction status will show that the deposit was successful.
 - ii. Else if the amount is less than 0, or balance is greater than absolute amount value, it indicates that the user has withdrawn money from the account. The balance will be updated accordingly, and transaction status will show successful withdrawal.
 - iii. Else, the transaction would not be successful due to insufficient funds, and the output string will indicate using string values.
2. The new balance of the userID will be saved for future references.
3. The transaction status will be returned.

Use Case 5: User Adds Item for Sale/Rent

Input:

- UserID (int): represents the unique ID for all approved users
- itemData (object): will hold value for all details stored within an item, including its description, price, and whether the item will be on rent or sale.

Output:

- listingStatus (string): will indicate if the item was successfully listed within the bidding system or not.

Pseudocode for Functionality:

1. Initially, it would be necessary to check if the user is a verified user to add items for sale/rent.
 - a. If not: Access will be denied, and the output will show accordingly.
2. Then, validate item data using the itemData function.
 - a. If not validated: Return a string indicating the Item Data was Invalid, and let the output show accordingly.
3. A new listingID will be created where the userID and itemData will be stored.
 - a. If the listingID then exists, a string value will represent a successful listing.
 - b. If not, it will indicate that the listing failed.
4. The status of the listing will be returned.

Use Case 6: User Bids on Items

Input:

- UserID (int): represents the unique ID for all approved users
- itemID (int): represents the unique ID for all successful listings, as derived from the previous use-case.
- bidAmount (float): represents the value which the user wishes to bid for a certain item.

Output:

- bidStatus: will indicate if the bid was successfully placed for an item or not.

Pseudocode for Functionality:

1. Initially, it would be necessary to check if the user is a verified user to bid an item.
 - a. If not: Access will be denied, and the output will show accordingly.
2. Next, the userBalance will be retrieved using the getBalance function created in the previous user case, called with the userID.
 - a. If the userBalance is less than the bidAmount - the code will stop and output insufficient funds.
3. A variable item will be declared using the accessor method to call the itemID.
 - a. If the item status is not available, or is None in the database - the code will output that the item is not available for bidding, and exit the code accordingly.
4. The BidID will be saved using a saveBid method, and the userID, itemID, and the bidAmount.
 - a. If the BidID was created successfully, it refers to a successful bid. The code will output that the bid was successfully placed.
 - b. Else: The bid failed, and the code will output accordingly.
5. The status of the bid placement will be returned.

Use Case 7: Transaction Ratings

Input:

- Transaction completion status: A completed transaction (purchase or rental).
- Rating: U provides a rating from 1 to 5 stars for the transaction.
- User: The U must be logged into their account.

Output:

- Confirmation message: The system confirms the rating submission.
- Error message: The system notifies U if there are the following problems: the transaction not being completed or the rating already submitted.

Pseudocode for Functionality:

1. User logs into their account
IF user is logged in THEN
 2. Check if the transaction is completed
IF transaction is completed THEN
 3. Display rating option (1 to 5 stars) to user
 4. User selects a rating (1 to 5 stars)
 5. Submit rating anonymously to the system
 6. System records the rating
 7. Display confirmation message: "Rating submitted successfully"
 - ELSE
 8. Display error message: "Transaction not completed"
- ELSE
 9. Display error message: "Please log in to submit a rating"

Use Case 8: User Complains about Another User

Input:

- User Login Status: (True/False)
- User provides a detailed complaint form specifying the issue.
- The transaction that the user is complaining about.

Output:

- A confirmation message notifying the user that their complaint has been successfully submitted and will be reviewed.
- An error message indicating issues such as incomplete form, ineligible transaction, or unmet complaint criteria.
- A notification sent to the system administrator (S) for review of the complaint.
- The action taken by the administrator after reviewing the complaint (e.g., warning, suspension, or further investigation).

Pseudocode for Functionality:

1. User logs into their account
IF user is logged in THEN
 2. User navigates to the complaint list of their recent transactions
 3. User selects a transaction to complain about
 4. User fills out the complaint form detailing the issue
IF complaint form is complete AND transaction is eligible THEN
 5. Submit the complaint form
 6. System sends a receipt of the complaint and notifies user that it will be reviewed
 7. System notifies the administrator (S) of the complaint
 - ELSE
 8. Display error message: "Incomplete complaint form" OR "Transaction not eligible for complaint" OR "Complaint criteria not met"
 - ELSE
 9. Display error message: "Please log in to submit a complaint"
2. Administrator reviews the complaint
IF administrator reviews complaint THEN
 10. Administrator takes appropriate action based on findings

Use Case 9: Suspension and Re-Activation

Input:

- User's average rating from other users after transactions (1-5 stars)
- Complaints or violation logs that trigger suspension.
- Fine payment or administrator review decision for reactivation.
- Suspension reasons and associated user information.

Output:

- Message notifying the user of their suspension, including the reason and how to reactivate the account.
- A record of the suspension is created and stored in the system.
- Message confirming that the user's account has been reactivated.
- Message notifying the user that their reactivation request has been denied, along with reasons.

- Message indicating an issue with the reactivation process, such as incomplete payment or missing information.

Pseudocode for Functionality:

1. System checks the user's average rating after a transaction
IF average rating is below 2 OR above 4 AND evaluated by at least 3 users THEN
 2. Flag the user for suspension
 3. Create a suspension record in the system
 4. Notify the user of suspension (reason and reactivation process)
 5. Disable user's access to functionalities (e.g., bidding, listing items)
2. User requests reactivation
IF user requests reactivation by paying the fine OR requests review by administrator THEN
 6. IF fine is paid THEN
 7. Confirm fine payment
 8. Notify the administrator to review the case
 - ELSE IF administrator review is requested THEN
 9. System administrator reviews the case
 10. Administrator approves or denies reactivation based on findings
IF reactivation approved THEN
 11. Reactivate the user's account and restore access
 12. Notify the user that the account has been reactivated
 - ELSE
 13. Deny reactivation and notify the user with reasons
 - END IF
 - ELSE
 14. Notify the user of missing requirements (e.g., fine not paid, incomplete information)

Use Case 10: Forced Removal of User

Input:

- List of suspension records, each containing a reason and timestamp.
- Current status of the user's account (whether active or suspended).
- Notifications sent to the user regarding previous suspensions and consequences.
- User's suspension count (must be three suspensions).
- Data about the forced removal event, including user ID, removal reason, and timestamp.

Output:

- A message sent to the user informing them of the forced removal, including the reasons and account status.
- The user's account is deactivated, and access to platform functionalities is revoked.
- A system record documenting the forced removal of the user, including user ID, reason for removal, and timestamp.
- Inform the user if there is an error in the system processing.

Pseudocode for Functionality:

1. System checks if the user has been suspended three times
IF user has been suspended 3 times THEN
 2. Verify the suspension records for accuracy
 3. IF suspension records are valid THEN
 4. Notify the user of the forced removal, including reasons and account status
 5. Deactivate the user's account, revoking all access and functionalities
 6. Create an audit record of the forced removal (user ID, reason, timestamp)
 - ELSE
 7. Notify the user of an issue with suspension records (e.g., "Suspension record error")
 8. Provide instructions to contact support for further assistance
 - ELSE
 9. No forced removal needed (user hasn't reached 3 suspensions)

Use Case 11: User Requests to Quit the System

Input:

- Current status of the user's account (whether active, suspended, or removed).
- The user's action to initiate a quit request from account settings.
- List of any active transactions that the user has not completed.
- A check for whether the user's request complies with system policies (e.g., no ongoing transactions, no suspensions).
- Data about the deactivation request, including user ID, reason for quitting, and timestamp.

Output:

- A confirmation prompt displayed to the user after initiating the quit request.
- A message sent to the user confirming the deactivation of their account.
- A system record documenting the deactivation of the user's account, including user ID, request reason, and timestamp.
- Inform the user if there is an error in the system processing.

Pseudocode for Functionality:

1. User navigates to account settings and selects the option to quit the system
IF quit option selected THEN
 2. Display confirmation prompt to user
IF user confirms the request THEN
 3. Log the quit request in the system for processing
 4. Review the request for compliance with system policies
IF user has ongoing transactions THEN
 5. Notify user to resolve ongoing transactions before proceeding
 6. Terminate process
 - ELSE IF user has no ongoing transactions AND is not under suspension THEN
 7. Proceed with deactivation
 8. Notify user that their account is deactivated
 9. Create an audit record with deactivation details (user ID, reason, timestamp)
 - ELSE
 10. Notify user of issue (e.g., "Account deactivation failed due to suspension")
 11. Terminate process
 - ELSE
 12. Cancel the request process
- ELSE
 13. Exit process (no action taken)

User Case 12: VIP Status Achieved

Input:

- Current status of the user's account (active/inactive).
- The user's current balance in the system.
- A list of transactions completed by the user.
- A list of complaints or reports against the user.
- Predefined criteria for achieving VIP status (balance > \$5,000, more than 5 transactions, no complaints).
- Data about the user's status (VIP or regular) and activity.

Output:

- A notification sent to the user informing them that their account has been upgraded to VIP status and that they now have special privileges (e.g., a 10% discount).
- Exclusive features granted to the user (e.g., discount, access to exclusive features)
- Ongoing tracking of the user's VIP status based on their balance, transaction history, and complaint count.
- A message notifying the user that their VIP status has been revoked due to failure to meet the criteria (e.g., balance drop, transaction count decrease, or complaints).
- A record of the user's VIP status, including any updates.
- Inform the user if there is an error in the system processing.

Pseudocode for Functionality:

1. Continuously monitor user accounts for VIP eligibility
 2. Retrieve user's account balance, transaction history, and complaint history
 3. Check if user meets VIP criteria
IF balance > \$5,000 AND transaction count > 5 AND no complaints THEN
 4. Update user's account to VIP status
 5. Assign VIP privileges (e.g., 10% discount, exclusive access)
 6. Send notification to user confirming VIP status update (VIP privileges granted)
 7. Log audit record for VIP status update
 - ELSE
 8. Continue monitoring user activity for VIP eligibility
2. Continuously track user's activity
 3. IF user no longer meets at least two of the VIP criteria THEN
 4. Revert user's account to regular status
 5. Notify user that their VIP status has been revoked

6. Log audit record for VIP status reversion
3. Handle any errors or discrepancies
4. IF system error or failure to meet criteria detected THEN
 5. Notify user of issue (e.g., "Error in VIP status update")
 6. Provide guidance for resolving the issue (e.g., "Please contact support")

SmartBid	Version: 2.0
Software Requirements Specification	Date: 12/11/2024
<document identifier>	

5. Supporting Information

Software Requirements Specification has:

- Table of Contents

Memos & Notes:

- 10/07/2024: Allocate primary responsibilities and timeline; begin drafting mockups.
- 10/15/2024: Check in and finalize Report 1.
- 10/21/2024: Check in and finish base mockups, begin coding front-end while researching backend.
- 10/28/2024: Review mockups and divide and conquer the homepage pages and the dashboard pages.
- 11/04/2024: Begin Report 2 and continue coding the front-end and back-end for functionality.
- 11/11/2024: Continue working on Report 2 and the project.
- Note: There are currently no concerns regarding our team work.

GitHub & Figma Links:

- GitHub: <https://github.com/saanavig/SmartBid>

- Figma:

<https://www.figma.com/design/pdjiBdDFsIgs4kRqkPfS2N/CSC-32200?node-id=31-2&t=hHALu96zkGT4bHlz-1>