Report

On

Group Project

Automated Stock Exchange System

INST 5336: Python Programming
Fall 2024

Group: Backbencher

Members:

Bhuwan Bokati
Pawan Ojha

Automated Stock Exchange System

Introduction

The Automated Stock Exchange System is designed to simulate a stock exchange environment where users can create accounts, place bids, and make asks for stocks. The system validates inputs, manages stock transactions, and logs executed transactions. The application involves user interactions such as account creation, placing bids and asks, and executing transactions based on matching bids and asks.

System Architecture

The system is built in Python and utilizes a file-based approach for persistent storage. Below is an overview of the components involved in the system:

1. Account Creation: Users can create accounts by providing their first name, last name, and email. The system ensures that the email format is valid and that the fields are not empty.
2. Bidding and Asking: Users can place bids and asks for stocks. Each bid and ask includes a stock name, price, amount, and user ID. The system matches bids and asks based on price, and orders are executed if the conditions are met.
3. Transaction Execution: When a match is found between a bid and an ask, the transaction is executed, and the details are logged in an execution log file (execution_log1.txt).
4. File Management:
    a) Account file: Stores all user account information (accounts1.txt).
    b) Execution log: Logs all successful transactions (execution_log1.txt).

System Flow and Functionality

Account Creation

1. User Input: The system prompts the user to input their first name, last name, and email.
2. Validation: The system checks if the fields are empty or if the email format is invalid.
3. Account Registration: Once valid inputs are provided, the system generates a unique user ID and saves the account details in a file.

Bidding

1. User Input: The user provides the stock name, bidding price, amount, and user ID.
2. Validation: The system checks if the stock name contains only valid characters, if the price is positive, and if the amount is valid.
3. Execution: If the bid is valid, the system attempts to match it with existing asks.

Asking

1. User Input: The user provides the stock name, asking price, amount, and user ID.

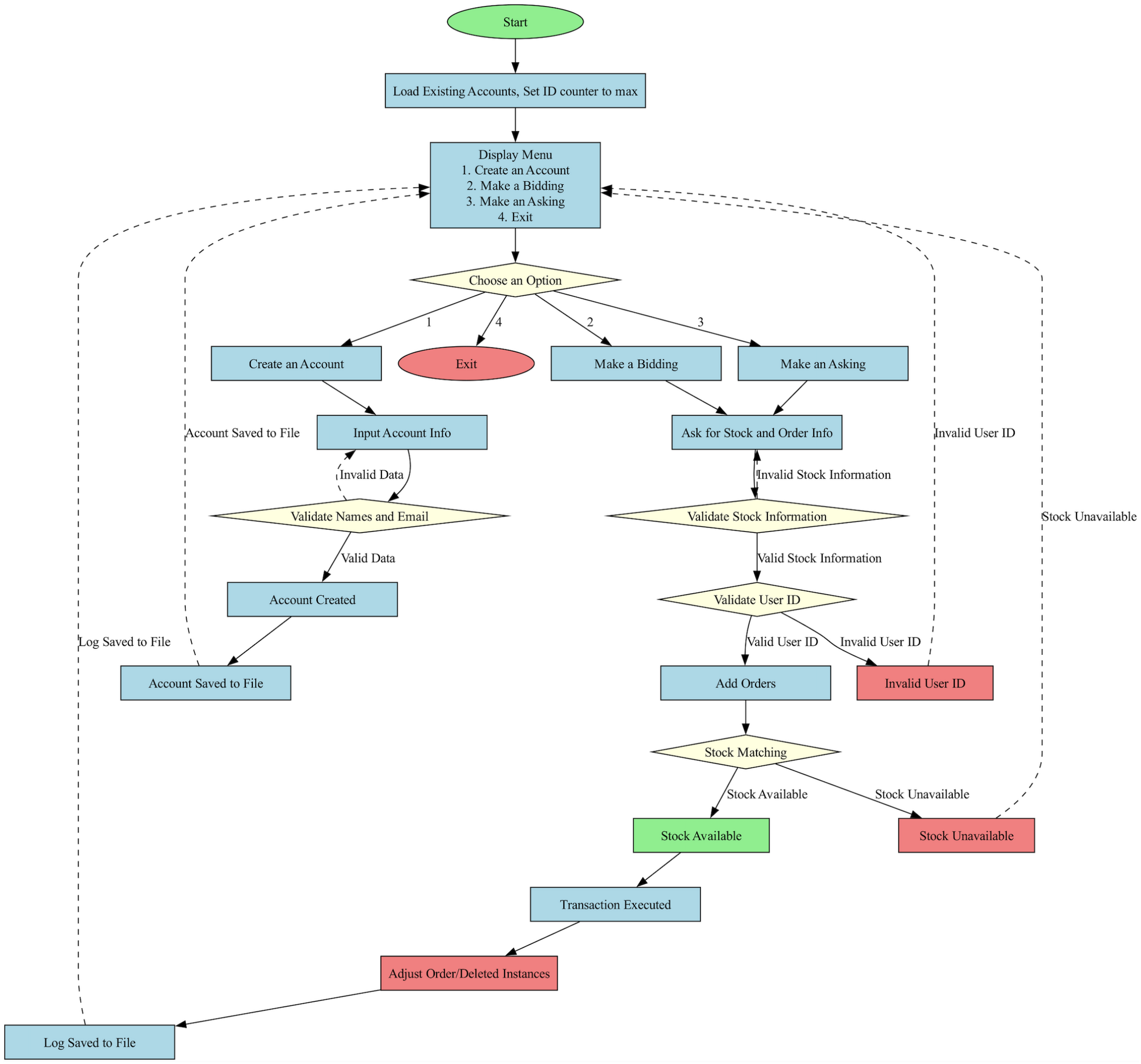
2. Validation: The system checks if the stock name is valid, the price is positive, and the amount is valid.
3. Execution: The system attempts to match the ask with existing bids.

Transaction Execution

1. Matching Logic: The system sorts the bids in descending order and asks in ascending order. If a bid meets or exceeds the ask price, a transaction is executed.
2. Logging: Once a transaction is executed, the details are logged in the execution log file.

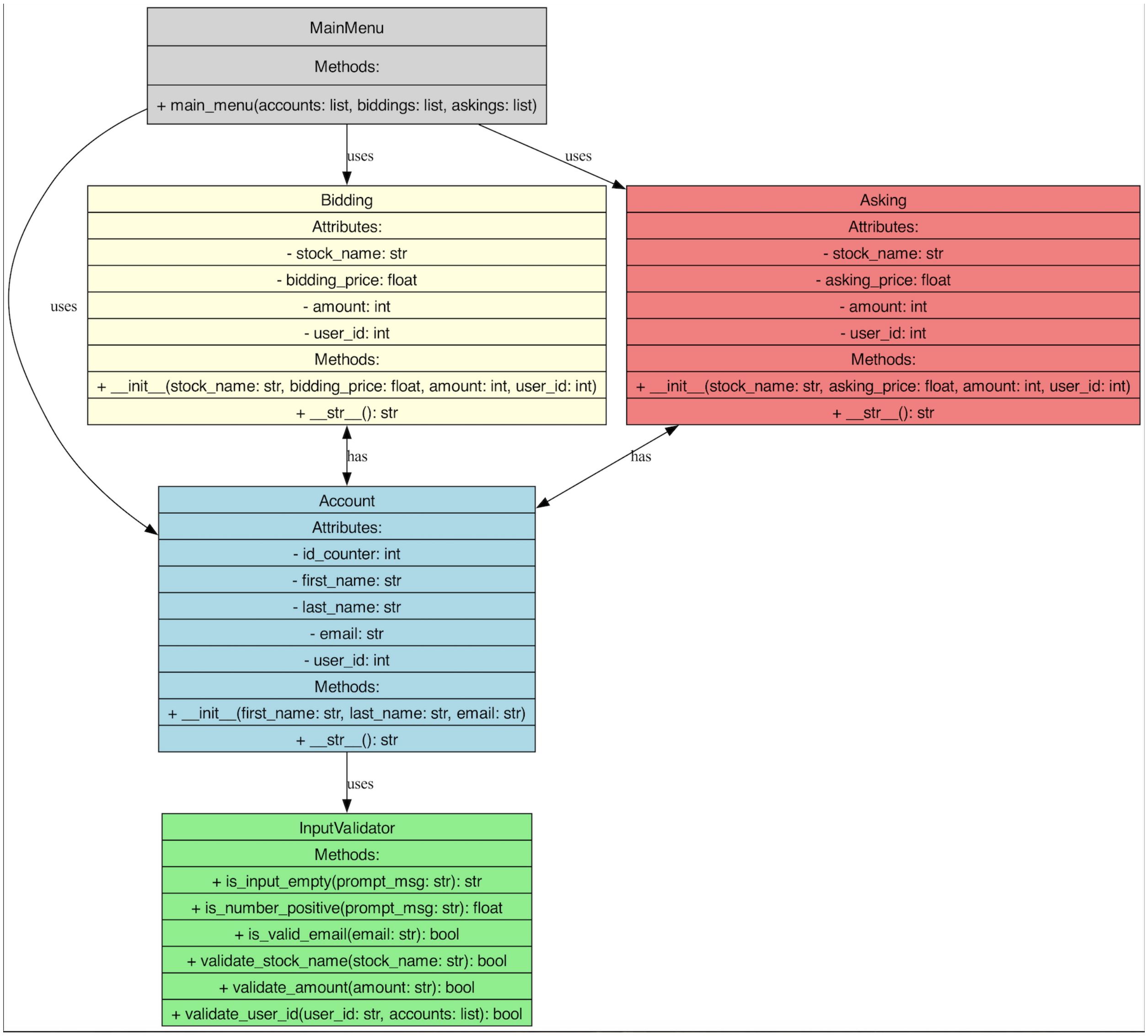
Flowchart

The flowchart for the Automated Stock Exchange System outlines the step-by-step process and decision-making logic involved in creating accounts, placing bids and asks, and executing transactions.



Unified Modeling Language (UML) Diagram

The below UML diagrams provide a clear representation of the system's structure and behavior, focusing on the relationships between the main components such as Account, Bidding, and Asking.


MainMenu
Methods:
+ main_menu(accounts: list, biddings: list, askings: list)
uses
uses
uses
Bidding
Attributes:
- stock_name: str
- bidding_price: float
- amount: int
- user_id: int
Methods:
+ __init__(stock_name: str, bidding_price: float, amount: int, user_id: int)
+ __str__(): str
Asking
Attributes:
- stock_name: str
- asking_price: float
- amount: int
- user_id: int
Methods:
+ __init__(stock_name: str, asking_price: float, amount: int, user_id: int)
+ __str__(): str
has
has
Account
Attributes:
- id_counter: int
- first_name: str
- last_name: str
- email: str
- user_id: int
Methods:
+ __init__(first_name: str, last_name: str, email: str)
+ __str__(): str
uses
InputValidator
Methods:
+ is_input_empty(prompt_msg: str): str
+ is_number_positive(prompt_msg: str): float
+ is_valid_email(email: str): bool
+ validate_stock_name(stock_name: str): bool
+ validate_amount(amount: str): bool
+ validate_user_id(user_id: str, accounts: list): bool

Testing Scenarios

Scenario 1: Valid Account Creation

- Input:

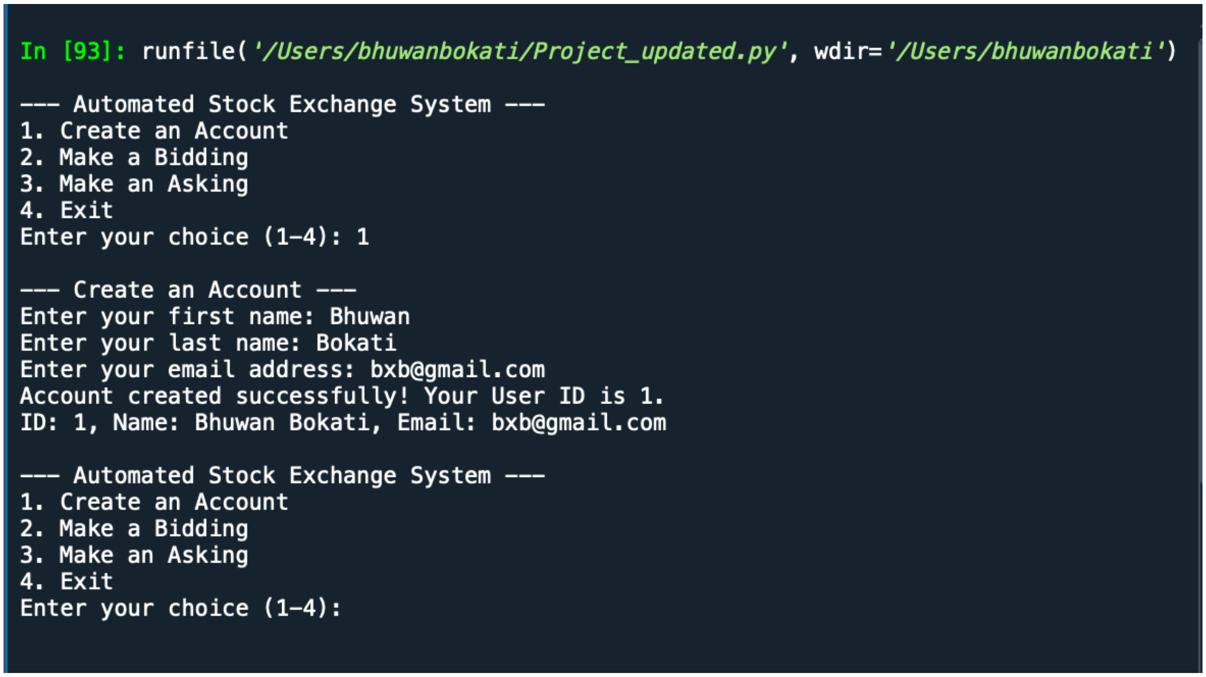
  First Name: "Bhuwan"

Last Name: “Bokati”
Email: “bxb@gmail.com”

- Expected Outcome:

  The account is created successfully with a unique user ID.
  The system saves the account information in accounts.txt.

Screenshot:



```
In [93]: runfile('/Users/bhuwanbokati/Project_updated.py', wdir='/Users/bhuwanbokati')

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 1

--- Create an Account ---
Enter your first name: Bhuwan
Enter your last name: Bokati
Enter your email address: bxb@gmail.com
Account created successfully! Your User ID is 1.
ID: 1, Name: Bhuwan Bokati, Email: bxb@gmail.com

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4):
```

Scenario 2: Invalid Email Format

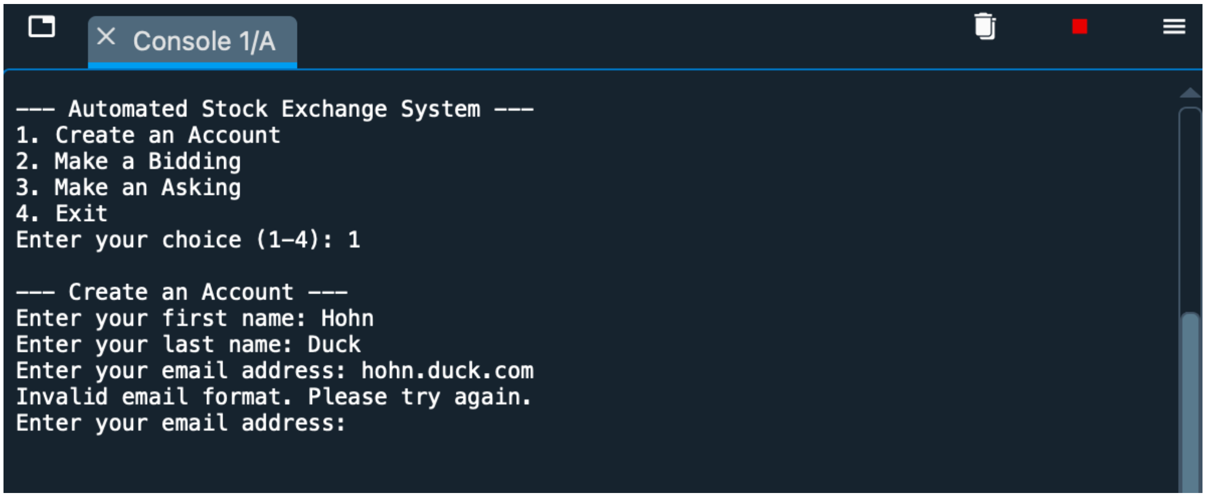
- Input:

  First Name: “Hohn”
  Last Name: “Duck”
  Email: “hohn.duck.com”(invalid format)

- Expected Outcome:

  The system prompts the user to re-enter the email until a valid one is provided.

Screenshot:



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 1

--- Create an Account ---
Enter your first name: Hohn
Enter your last name: Duck
Enter your email address: hohn.duck.com
Invalid email format. Please try again.
Enter your email address:
```

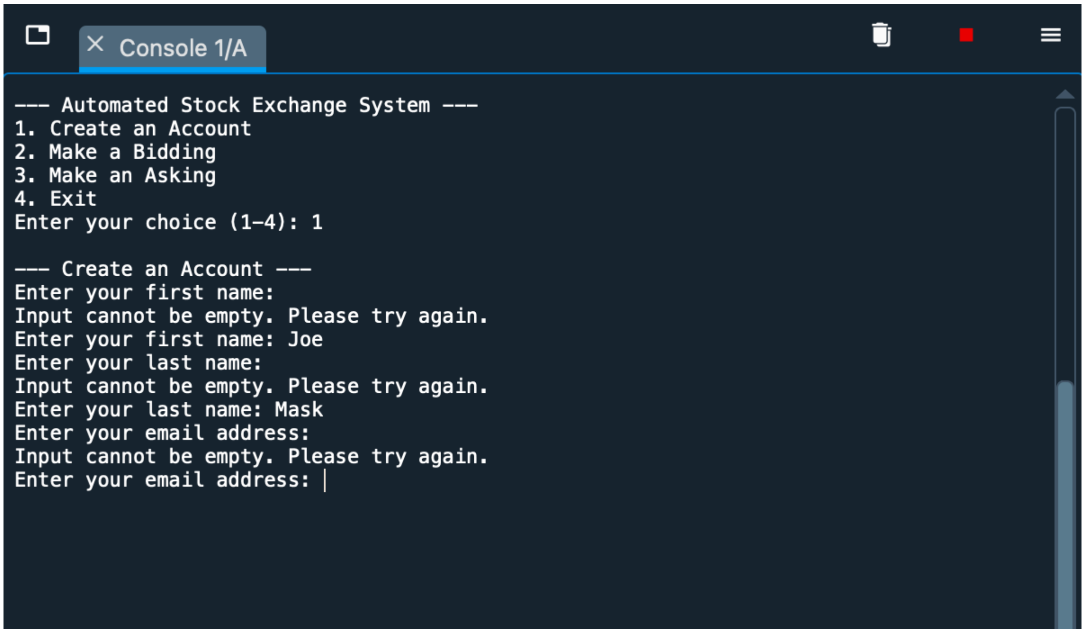
Scenario 3: Empty String

- Input:

  First Name: “ “
  Last Name: “ ”
  Email: “ ”

- Expected Outcome:

  The system prompts the user to re-enter inputs until a valid one is provided.



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 1

--- Create an Account ---
Enter your first name:
Input cannot be empty. Please try again.
Enter your first name: Joe
Enter your last name:
Input cannot be empty. Please try again.
Enter your last name: Mask
Enter your email address:
Input cannot be empty. Please try again.
Enter your email address: |
```

Scenario 4: Valid Bidding

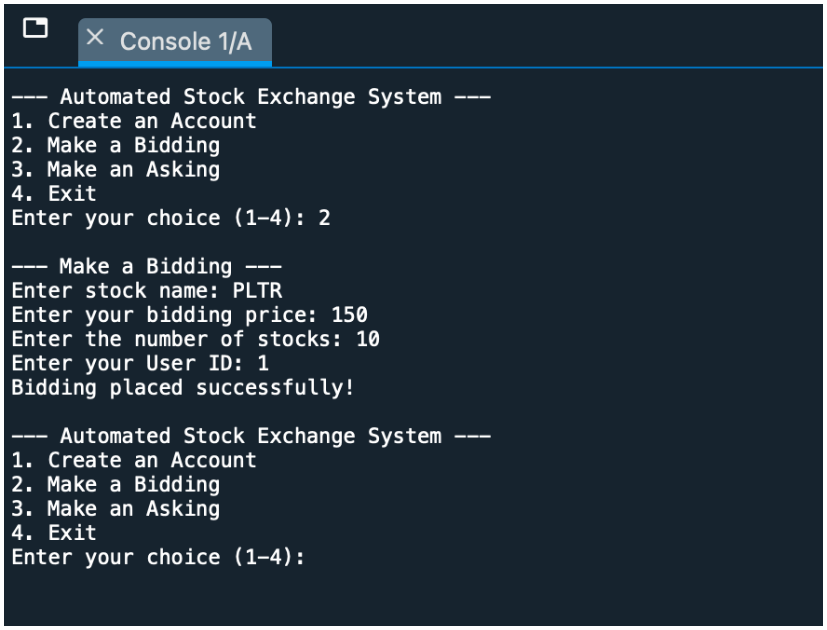
- Input:

  Stock Name: "PLTR"
  Bidding Price: 150
  Amount: 10
  User ID: 1

- Expected Outcome:

  The bid is placed successfully and stored in the system.
  If there is an ask that matches this bid, the system will execute the transaction.

Screenshot:



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 2

--- Make a Bidding ---
Enter stock name: PLTR
Enter your bidding price: 150
Enter the number of stocks: 10
Enter your User ID: 1
Bidding placed successfully!

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4):
```

Scenario 5: Invalid Bidding (Negative Amount)

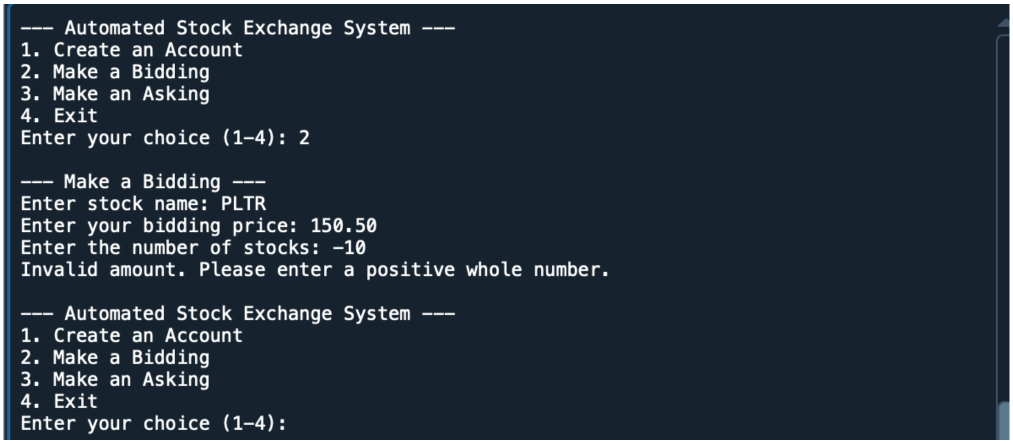
- Input:

  Stock Name: "PLTR"
  Bidding Price: 150.50
  Amount: -10 (invalid)
  User ID: 1

- Expected Outcome:

  The system prompts the user to enter a valid amount.

Screenshot:



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 2

--- Make a Bidding ---
Enter stock name: PLTR
Enter your bidding price: 150.50
Enter the number of stocks: -10
Invalid amount. Please enter a positive whole number.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4):
```

Scenario 6: Valid Ask

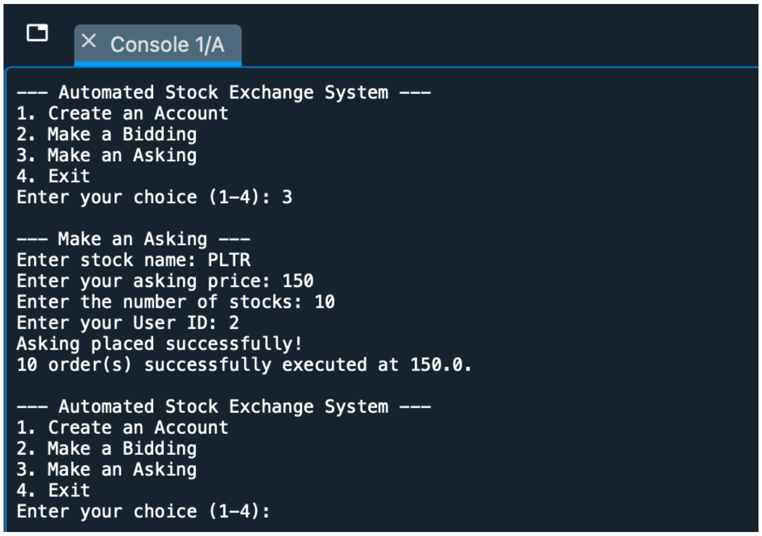
- Input:

  Stock Name: "PLTR"
  Bidding Price: 150
  Amount: 10
  User ID: 2

- Expected Outcome:

  The bid is placed successfully and stored in the system.
  If there is an ask that matches this bid, the system will execute the transaction.

Screenshot:



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 3

--- Make an Asking ---
Enter stock name: PLTR
Enter your asking price: 150
Enter the number of stocks: 10
Enter your User ID: 2
Asking placed successfully!
10 order(s) successfully executed at 150.0.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4):
```

Scenario 7: Invalid Ask (Negative Amount)

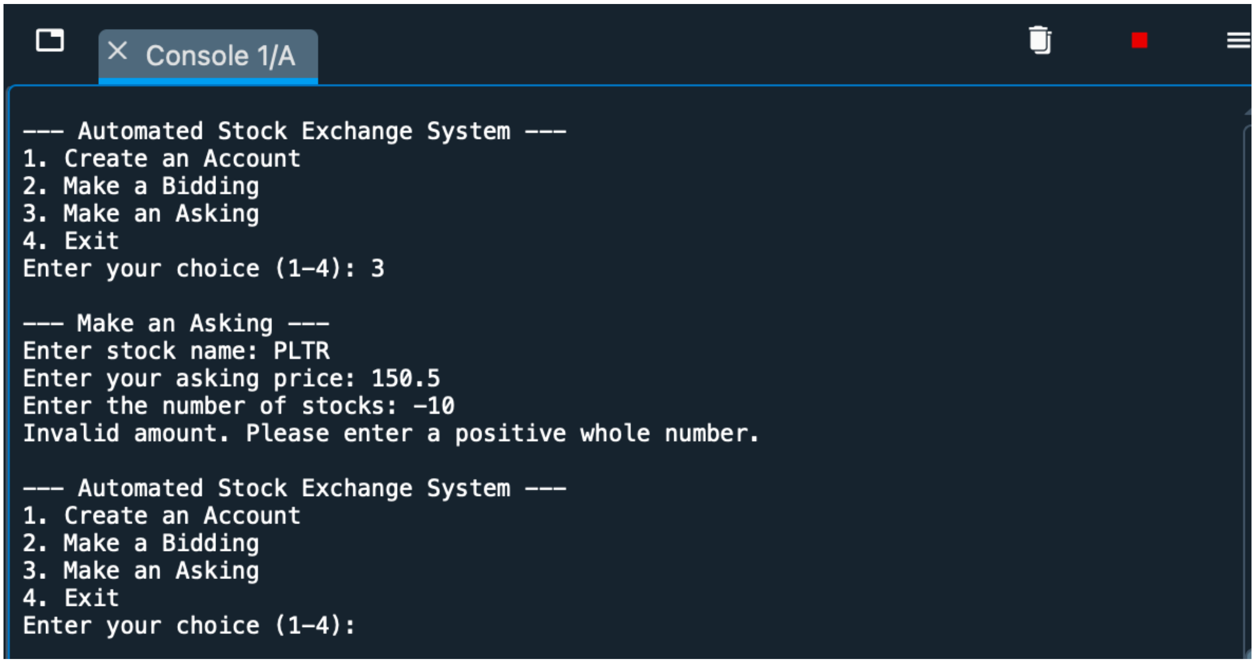
- Input:

  Stock Name: "PLTR"
  Bidding Price: 150.50
  Amount: -10 (invalid)
  User ID: 1

- Expected Outcome:

  The system prompts the user to enter a valid amount.

Screenshot:



```
--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 3

--- Make an Asking ---
Enter stock name: PLTR
Enter your asking price: 150.5
Enter the number of stocks: -10
Invalid amount. Please enter a positive whole number.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4):
```

Scenario 8: Transaction Matching (Successful Match)

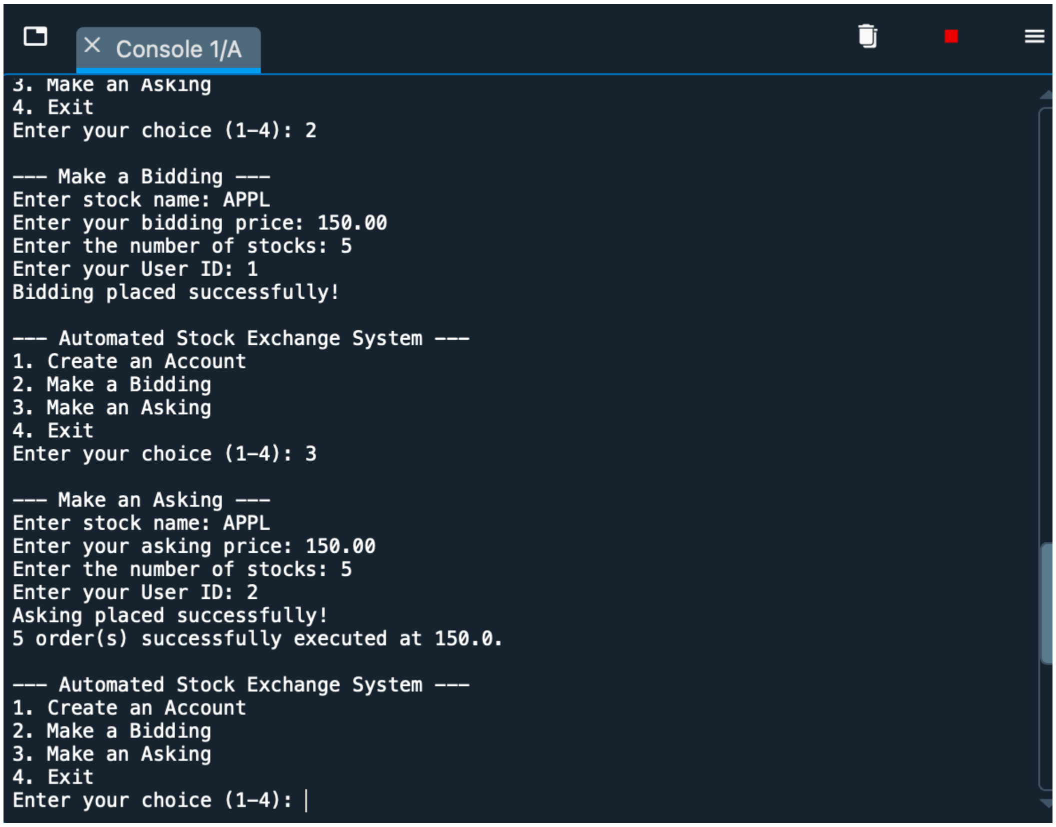
- Input:

  Place a bid for "AAPL" with price 150.50 and amount 5.
  Place an ask for "AAPL" with price 150.00 and amount 5.

- Expected Outcome:

  The system matches the bid and ask, executes the transaction, and logs it in the execution_log1.txt file.

Screenshot:



```
3. Make an Asking
4. Exit
Enter your choice (1-4): 2

--- Make a Bidding ---
Enter stock name: APPL
Enter your bidding price: 150.00
Enter the number of stocks: 5
Enter your User ID: 1
Bidding placed successfully!

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 3

--- Make an Asking ---
Enter stock name: APPL
Enter your asking price: 150.00
Enter the number of stocks: 5
Enter your User ID: 2
Asking placed successfully!
5 order(s) successfully executed at 150.0.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): |
```

Scenario 9: Invalid User ID

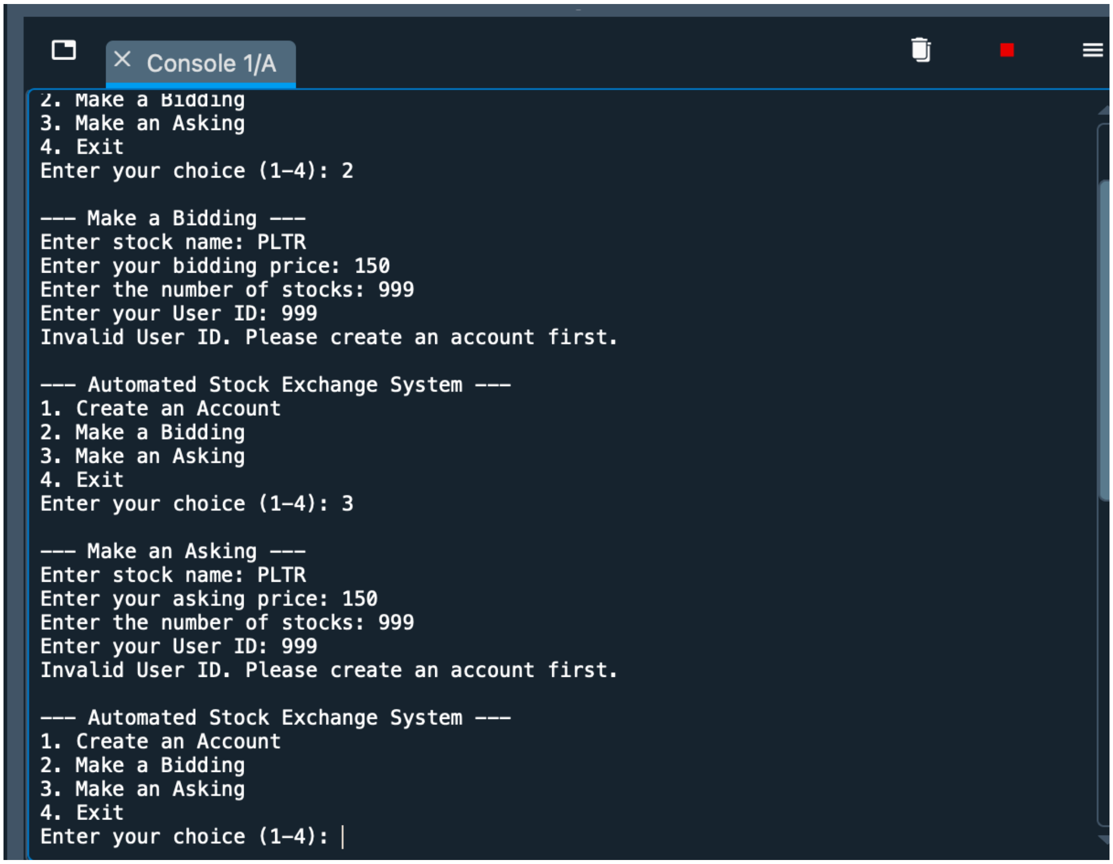
- Input:

  Stock Name: "PLTR"
  Bidding Price: 150
  Amount: 999
  User ID: 999 (non-existing)

- Expected Outcome:

  The system prompts the user to enter a valid user ID.

Screenshot:



```
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 2

--- Make a Bidding ---
Enter stock name: PLTR
Enter your bidding price: 150
Enter the number of stocks: 999
Enter your User ID: 999
Invalid User ID. Please create an account first.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 3

--- Make an Asking ---
Enter stock name: PLTR
Enter your asking price: 150
Enter the number of stocks: 999
Enter your User ID: 999
Invalid User ID. Please create an account first.

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): |
```

Scenario 10: Partial Biding and Asking Price less than Biding Price

- Input:

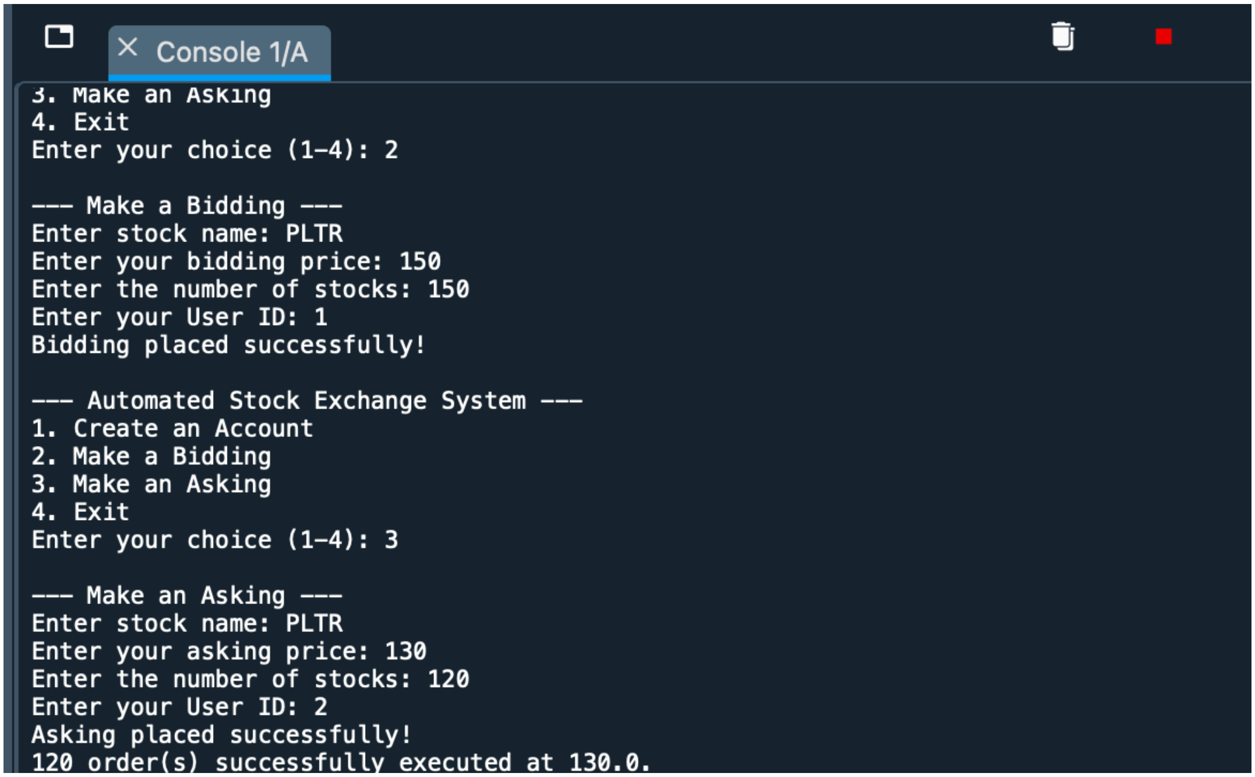
  Bidding
  Stock Name: "PLTR"
  Bidding Price: 150
  Amount: 150
  User ID: 1

  Asking
  Bidding
  Stock Name: "PLTR"
  Bidding Price: 130
  Amount: 120
  User ID: 2

- Expected Outcome:

  The partial bid amount will be fulfilled, and the pricing will be at asking price

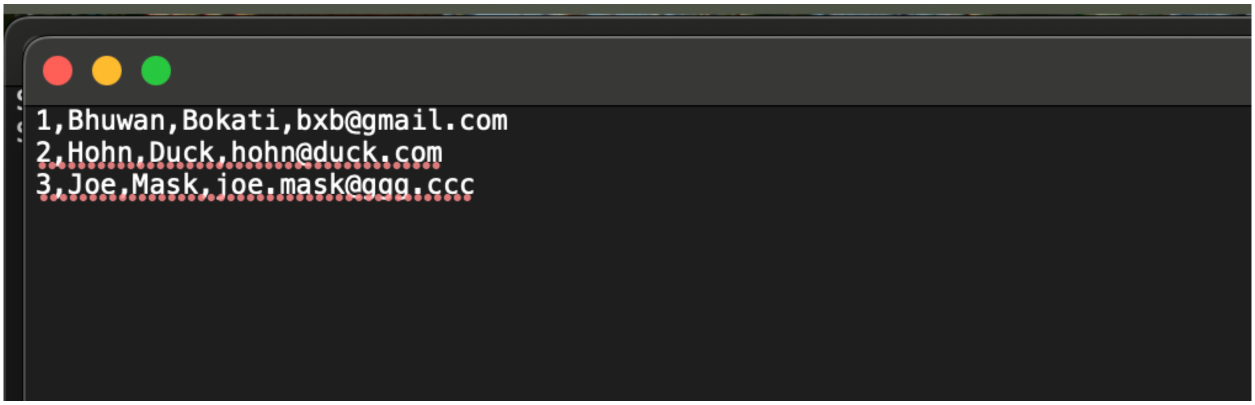
Screenshot:



```
3. Make an Asking
4. Exit
Enter your choice (1-4): 2

--- Make a Bidding ---
Enter stock name: PLTR
Enter your bidding price: 150
Enter the number of stocks: 150
Enter your User ID: 1
Bidding placed successfully!

--- Automated Stock Exchange System ---
1. Create an Account
2. Make a Bidding
3. Make an Asking
4. Exit
Enter your choice (1-4): 3

--- Make an Asking ---
Enter stock name: PLTR
Enter your asking price: 130
Enter the number of stocks: 120
Enter your User ID: 2
Asking placed successfully!
120 order(s) successfully executed at 130.0.
```

Account Creation File

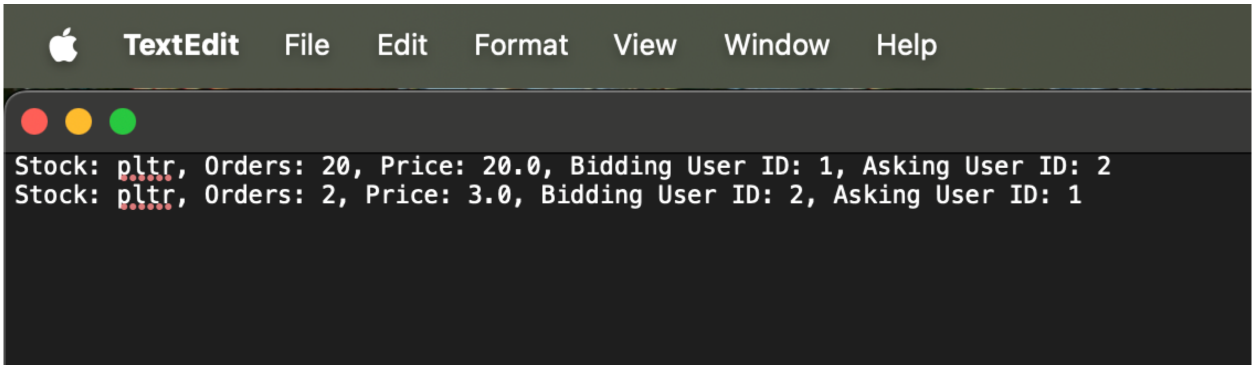
After a successful account creation is successful, the information is saved in a file and will look like this.



```
1,Bhuwan,Bokati,bxb@gmail.com
2,Hohn,Duck,hohn@duck.com
3,Joe,Mask,joe.mask@ggg.ccc
```

Execution Log File

After a successful transaction, the execution_log.txt file will look like this:



```
Stock: pltr, Orders: 20, Price: 20.0, Bidding User ID: 1, Asking User ID: 2
Stock: pltr, Orders: 2, Price: 3.0, Bidding User ID: 2, Asking User ID: 1
```

This log captures the stock name, the number of orders executed, the transaction price, and the user IDs involved in the transaction.

Team Contributions

Team Member 1 (Bhuwan Bokati):

- Responsible for the development of the Account class, Bidding and Asking classes.
- Implemented the input validation functions for user inputs (e.g., email validation, price validation).
- Designed the logic for transaction matching and execution, ensuring that the system executes matching bids and asks.

Team Member 2 (Pawan Ojha):

- Developed the file management system to save and load accounts from accounts.txt and log transactions in execution_log.txt.
- Worked on ensuring the proper handling of invalid inputs and error messages.
- Conducted testing of different scenarios to ensure system stability and functionality.

Conclusion

The Automated Stock Exchange System effectively simulates a basic stock exchange environment, allowing users to create accounts, place bids and asks, and execute transactions. Through input validation, matching logic, and transaction logging, the system ensures smooth operation while handling various user scenarios. The project was successfully developed through collaborative efforts, ensuring that all major functionalities were implemented and tested.