Computer and Network Security

ATM-BANK COMMUNICATION PROTOCOL

HIGH-LEVEL IMPLEMENTATION (22/09/24)

Group 1

Heer Kubadia (22110096)
Jiya Desai (22110107)
Lavanya (22110130)
Nishi Shah (22110171)
Pratham Sharda (22110203)
Harshita Singh (22110299)

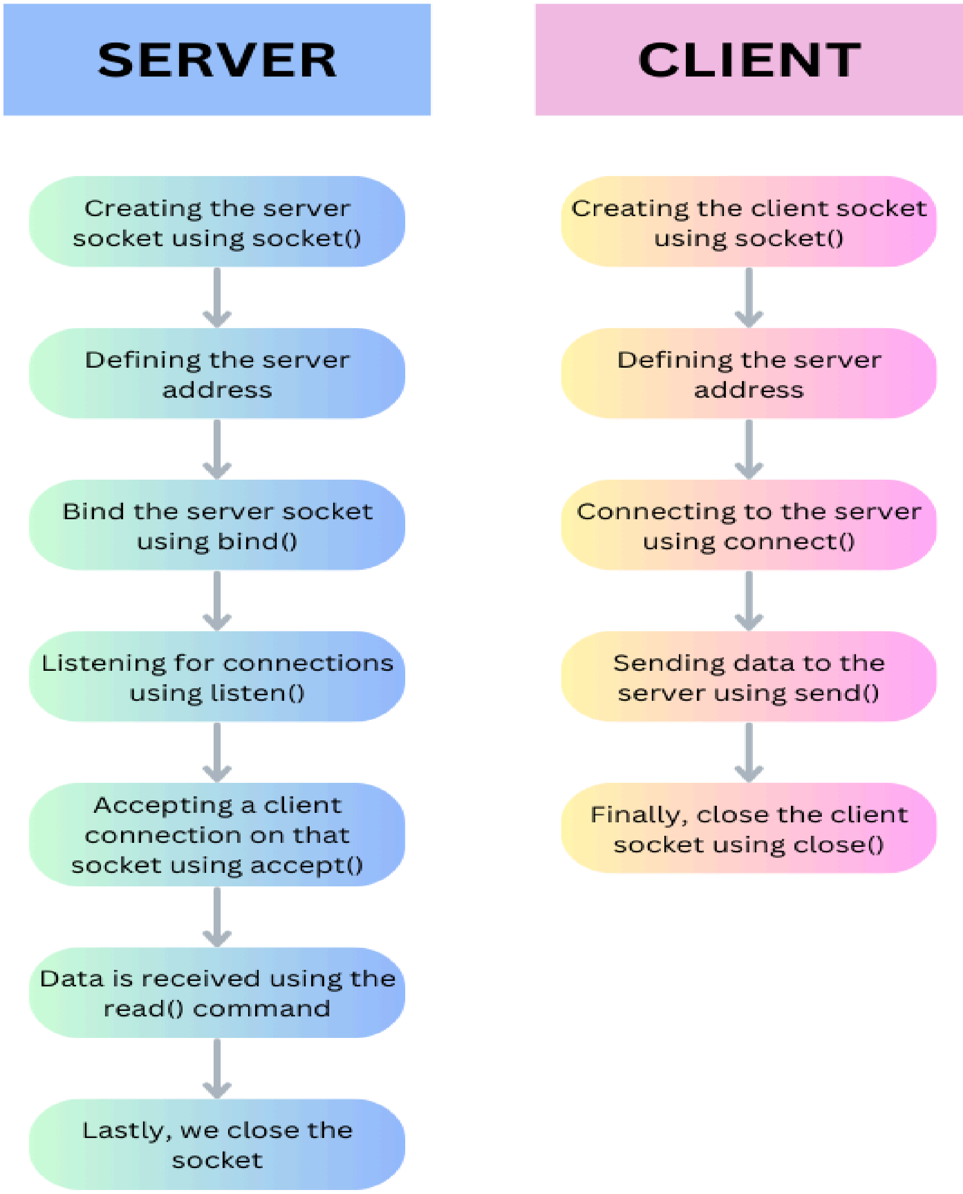
OVERVIEW OF ACHIEVEMENTS SO FAR

1. **Basic Client-Server Architecture**:
   - Successfully established a client-server architecture where the ATM client communicates with the bank server over TCP sockets. Implemented networking functionalities for establishing connections, sending requests, and receiving responses, ensuring reliable communication between components.
2. **Command Implementation**:
   - Implemented core commands such as account registration, login, deposit, and withdrawal, allowing users to perform basic banking operations.
   - These operations can only be performed IF you are logged in, otherwise, 'Not logged in' will show up as an error message.
3. **JSON Communication**:
   - Utilized JSON formatting for requests and responses, enabling structured communication between the client and server.
4. **Session Management**:
   - Developed a session management mechanism that generates session tokens upon successful login, which will be useful for ensuring that sensitive operations like deposits and withdrawals are only allowed for authenticated users.
5. **Error Handling**:
   - Incorporated basic error handling for various scenarios, including invalid commands and login checks, improving the robustness of the system.

6. **Data Storage**:
   - Initiated a basic storage mechanism (CSV files) for user data and transactions for persistent storage.
7. **Modular Design**:
   - Created distinct modules for both the ATM client and bank server, facilitating separation of concerns and making the system easier to manage and extend.

HIGH LEVEL IMPLEMENTATION

1. Client-side and server-side socket creation sequence


SERVER
Creating the server socket using socket()
Defining the server address
Bind the server socket using bind()
Listening for connections using listen()
Accepting a client connection on that socket using accept()
Data is received using the read() command
Lastly, we close the socket
CLIENT
Creating the client socket using socket()
Defining the server address
Connecting to the server using connect()
Sending data to the server using send()
Finally, close the client socket using close()

2. **Unique user session creation**: The unique user session ID is a mechanism used to track and manage the session of a logged-in user, ensuring that the user remains authenticated throughout their interaction with the system (e.g., during ATM transactions). It is analogous to creating an instance of a unique temporary ID issued to the current user. The session remains active until the user chooses to exit the session. This is the first level of authentication in our model.

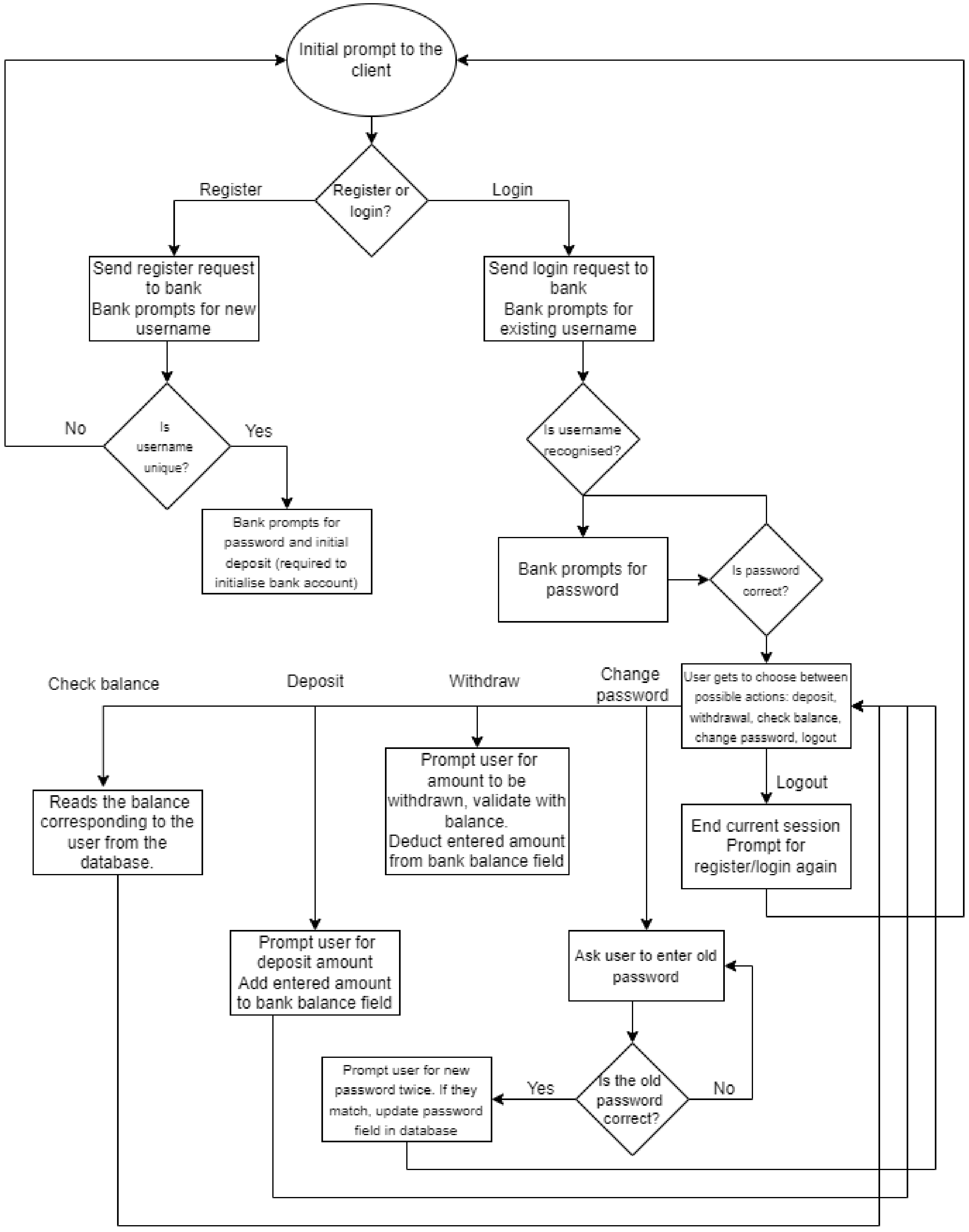
3. **ATM (client) functionalities** - sending requests to the bank (server)
    - Register (database entry creation - initialized with initial deposit amount)
    - Login (verifies existing user)
    - Check bank balance
    - Deposit amount
    - Withdraw amount
    - Change password (optional)
    - Exit

    But the functionalities related to monetary transactions can only be accessed once you are successfully logged in.

4. **Bank (server) functionalities** - receiving the client's request, database access, responding to the request
    - Register request:
      Once, it receives a new username - verifies after checking the database if the username is unique-
        - if not, ask user to choose a new action
        - if unique - prompt for password and initial deposit

      Other optional details include age, address, DOB, etc.
      So far in our model, the unique identification aspect is only the username
    - Login request:
      Receives username first:
      If username exists in the database, then ask for password
      If the password matches, send a 'Login successful' message and create a unique session for the user. During that session, depending on the action chosen by the user (deposit/check balance/withdraw/change password), the database will be accessed and updated, IF the unique session token is verified on the server and client side. Then, relevant message will be sent to the user.
    - Logout request: end the session, display initial prompt (register/login) again

OVERVIEW OF THE INTENDED MODEL


Initial prompt to the client
Register or login?
Register
Login
Send register request to bank
Bank prompts for new username
Send login request to bank
Bank prompts for existing username
Is username unique?
No
Yes
Bank prompts for password and initial deposit (required to initialise bank account)
Is username recognised?
Bank prompts for password
Is password correct?
User gets to choose between possible actions: deposit, withdrawal, check balance, change password, logout
Check balance
Deposit
Withdraw
Change password
Logout
Reads the balance corresponding to the user from the database.
Prompt user for amount to be withdrawn, validate with balance.
Deduct entered amount from bank balance field
End current session
Prompt for register/login again
Prompt user for deposit amount
Add entered amount to bank balance field
Ask user to enter old password
Is the old password correct?
Yes
No
Prompt user for new password twice. If they match, update password field in database

TASKS FOR THE NEXT PHASE OF THE PROJECT:

1) Currently, the code can only handle one action when we run the code, we will ensure that multiple requests from the user can be accommodated.
2) We will add more security features and authentication in addition to the unique user session token in which we must ensure that the same token ID cannot be reused once a session has ended or expired.
3) Change password, deposit, withdraw, etc. functionalities (working in tandem with the CSV file data) need to be added.
4) Currently, all the actions are displayed together. But ideally, on startup, only 3 options should be visible to the user. These are:
    a) Register
    b) Login
    c) Exit

    Once successfully logged in, the next set of functionalities will be shown. These are:
    a) Check bank balance
    b) Deposit amount
    c) Withdraw amount
    d) Change password (optional)
5) A makefile and authentication file must be made.