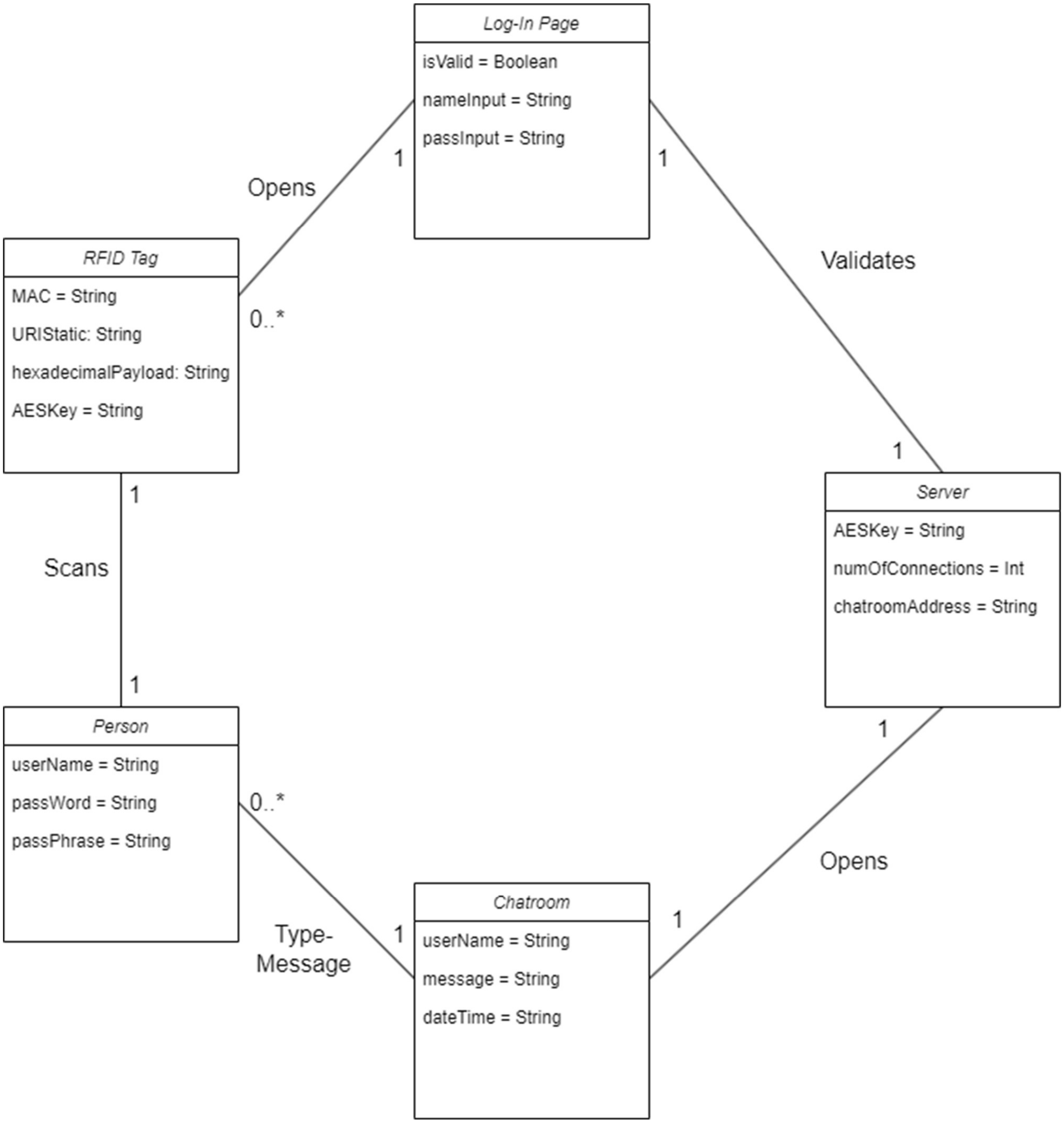
Domain Model Refinement and Analysis

1. Refine Domain Model



2. Class and Attribute Analysis

- *Server* Class- The server hosts the login page and chatroom while also managing user connections (threads) and encrypting messages in real time.
    - "AESKey" Attribute: This is used to encrypt messages through AES cyphering. AES 128 or 256 will suffice but encryption is needed to ensure that messages cannot be eavesdropped on.
    - "NumOfConnections" Attribute: This will allow the server to see how many people are connected at once and support developers to properly manage the application.
    - "chatroomAddress" Attribute: This is the link that will redirect user to the chat room If the user's login information has been validated.
- *Person* Class- The person is the actual user and device used to access the chat room. *Person* will have to enter the passphrase to actually utilize the RFID tag. *Person* will then enter their password into the login page in order to gain access to the chat room.
- *RFID Tag* Class: This is the physical verification token required to access the login page. The RFID tag acts as a username to a password. You can have "a" password but without the username that corresponds to the password, the password is useless. The AES key is also on the tag to decrypt the messages that have been encrypted on the chatroom.
- *Log-In Page* Class- This is where the user is brought after they tap the RFID tag. It is similar to a traditional log-in page, Facebook log-in for example. The RFID enabled chatroom log-in does not require a username as the RFID tag acts as the username. The only item that the log-in requests is a password (as described in the RFID Tag Class section).
- *Chatroom* Class- This is where the *Person*'s (users) will securely communicate with each other.

3. Association Analysis

The 3 associations that we are going over are Scans, Opens, and Validates between the *Person* and *RFID Tag*, *RFID Tag* and *Log-in Page*, and Log-in Page and Server.

1. Association: Scans; Classes: *Person* and *RFID Tag*

Person will have a unique RFID tag that is encoded with a hexadecimal representation of a username, MAC, URI, password/passphrase. When the user wishes to enter the chatroom, they will scan the RFID tag with a reader (smartphone) and enter a passphrase to move on to the next step.

2. Association: Opens; Classes: *RFID Tag* and *Log-in*

   After *Person* scans the *RFID tag* and inputs a correct passphrase, a link with populate the reader interface (phone). *Person* will OPEN or click the link to be directed to the web interface that is our log-in page.

3. Association: Validates; Classes: *Log-In Page* and *Server*
   After Person is brought to the log-in page, they are prompted to enter the password associated with the account while the username is automatically populated from the *RFID Tag*, which is then validated by a database and sent to the server to create a thread for the user.

4. Potential Methods

- *RFID Tag*
  - Method 1- decryptMessage()
  - Method 2- openLoginAddress()
- *Login Page*
  - Method 1 – getUsername()
  - Method 2 – checkPassword()
- *Server*
  - Method 1- createThread()
  - Method 2- encryptMessage()
- Chatroom
  - Method 1- displayMessage()
  - Method 2- displayTime()
- Person
  - Method 1 – sendMessage()
  - Method 2 – closeChatroom()

5. Reflection

Our domain model has gradually improved in both functionality and structure since it was first developed in class. Initially, the classes were *Person*, *RFID Tag, Server*, *Chatroom*, and *Database*. However, the Database is not meant to be shown in this model as it is a software artifact. Another change was the addition of the *Login Page,* which is an essential part of the program. One of the major aspects we missed when creating a domain model in class was the associations between the classes. As we progressed, we realized the need to clarify the interactions between these classes, by specifying how they connect. Improving attributes, particularly in terms of authentication and encryption, also took place. Important steps included the addition of attributes such as “AESKey” and methods to handle encryption and decryption of messages.

Ultimately, our domain model now better supports the product's vision of secure, authenticated communication, with each class working cohesively to ensure user verification, secure message transmission, and real-time interaction in the chatroom.