Chat Room CLI Client LLD

Terminology

Chat Room

A virtual environment in which users can post their messages and read the messages written by other users.

User

A person who interacts with the system.

Nickname

A familiar or humorous name the user uses to identify himself.

Registration

The act of recording user details.

Login

The act of signing into the system by the user.

Message

The text which the user delivers. Message content is limited to 150 characters.

Message Frame

A written communication sent between the users of the system.

Buissness Layer

Functionality

Performs all logic functions..

ChatRoom class

Functionality

Main client class. Maintains and operates all functionality methods.

Fields

- loggedInUser : User – current logged in user.
- url : string – server url.
- messages : List<IMessage> - contains all sent / retrieved messages
    - Sorted by timestamp.
    - Synched with persistent layer messages.
- users : List<User> - contains all registered users.
    - Sorted by g_id and lexicographic nickname order.
    - Synched with persistent layer users.
- request : Request – active request class.

Methods

- ChatRoom()
    - Initiates all fields
    - Synchronizes users and messages with persistent data.
- login(int g_id, string nickname) : bool
    - verifys valid user details.
    - Changes logged in user
    - Returns true if successful.
- logout(): void
    - Changes logged in user
- exit() : void
    - logs out.
    - Closes program.
- register(int g_id , string nickname) : void
    - creates new user.
    - Saves user data to users and to persistant layer
- retrieveMessages(int number): void
    - gets specified number of last messages from sever.
    - Saves messages to messages and to persistent layer.
- send(string message)
    - request.send(message,loggedInUser)
    - saves IMessage to messages and persistent layer.
- displayNMessages(int num) : List<IMessages>

    - returns num IMessages from messages.
- displayUserMessages(int g_id, string nickname) : List<IMessages
    - returns all IMessages sent by specified user details.
    - Saves messages to messages and to persistent layer.

User class

Functionality

User class, contains user details. Serializable.

Fields

- nickname : string
- g_id : int.

Request Class

Functionality

In charge of all requests from communication layer. Makes sure not to flood server with more than 20 queries in 10 seconds.

Fields

- final MAX_MESSAGE_LENGTH : int
- final URL : string
- lastNRequests : Queue
    - contains DateTime values of last N_ALLOWED queries sent.
    - Used to make sure not to overload server with more than N_ALLOWED queries in N_SECS
- Final N_ALLOWED : int
- Final N_SECS : int

Methods

- send(string msg, User user) : IMessage
    - validates msg.
    - makes a send request to comm' layer.
    - returns IMessage retrieved from comm' layer.
- retrieveMessages(int num) : List<IMessage>
    - makes a retrieve request to comm' layer.
    - returns List<IMessage> of num IMessages retrieved from comm' layer.
- isOverloading() : bool
    - returns true if sending another request will be over N_ALLOWED requests in N_SECS.
    - If true is returned request shouldn’t be sent.

- private validateMessage(string msg) : bool
    - validates msg.
    - returns true if valid.

Presentation layer – CLI

MainMenu class

Functionality

Manages the I/O with the chat user.
Communicates with the Buissness layer for which action the program has to take.

Fields

- userInput: int
  - The variable gets his value from the user (CLI) in order to know what kind of function/action the user wants to take.
- text: String
  - The content (the body of the message, the user's ID, nickname etc).

Methods

- getInput(): String
  - In charge of reciveing the chats user's text (e.g. userID/ nickname).
- printOutput(String text): Boolean
  - Prints text to the CLI (e.g. "login successful", "invalid message").
  - Returns true iff text printed successfully.
- toBuissness(int userInput): void
  - Directs to the specific function in the Buissness layer the client wants to take.

Persistent Layer

Functionality

Maintains persistent data regarding the client in local files.

Files

Stored in a local folder. The path is static.

Log4net package

Functionality

Creates and manages logging for the entire client

Handler<T> interface

Functionality

Saves, edits and retrieves data from a given type in a database.

Functions

- save(T data): bool
    - Saves data in the files system
    - Returns true if the data was saved successfully
- edit(T data): void
    - Edits data in the files system
    - Returns true if the data was edited successfully
- retrieveAll(): List<T>
    - Retrives all the data from type T in the files system

MassegeHandler<CommunicationMessage>

Functionality

Implements Handler. Manages persistency for *CommunicationMessage*.

Fields

- path: final static string
    - The path to the local folder containing the persistent data

UserHandler<User>

Functionality

Implements Handler. Manages persistency for *User*.

Fields

- path: final static string
    - The path to the local folder containing the persistent data

Communiction Layer

Functionality

Performs all communication with server. Given as an outsourced layer.

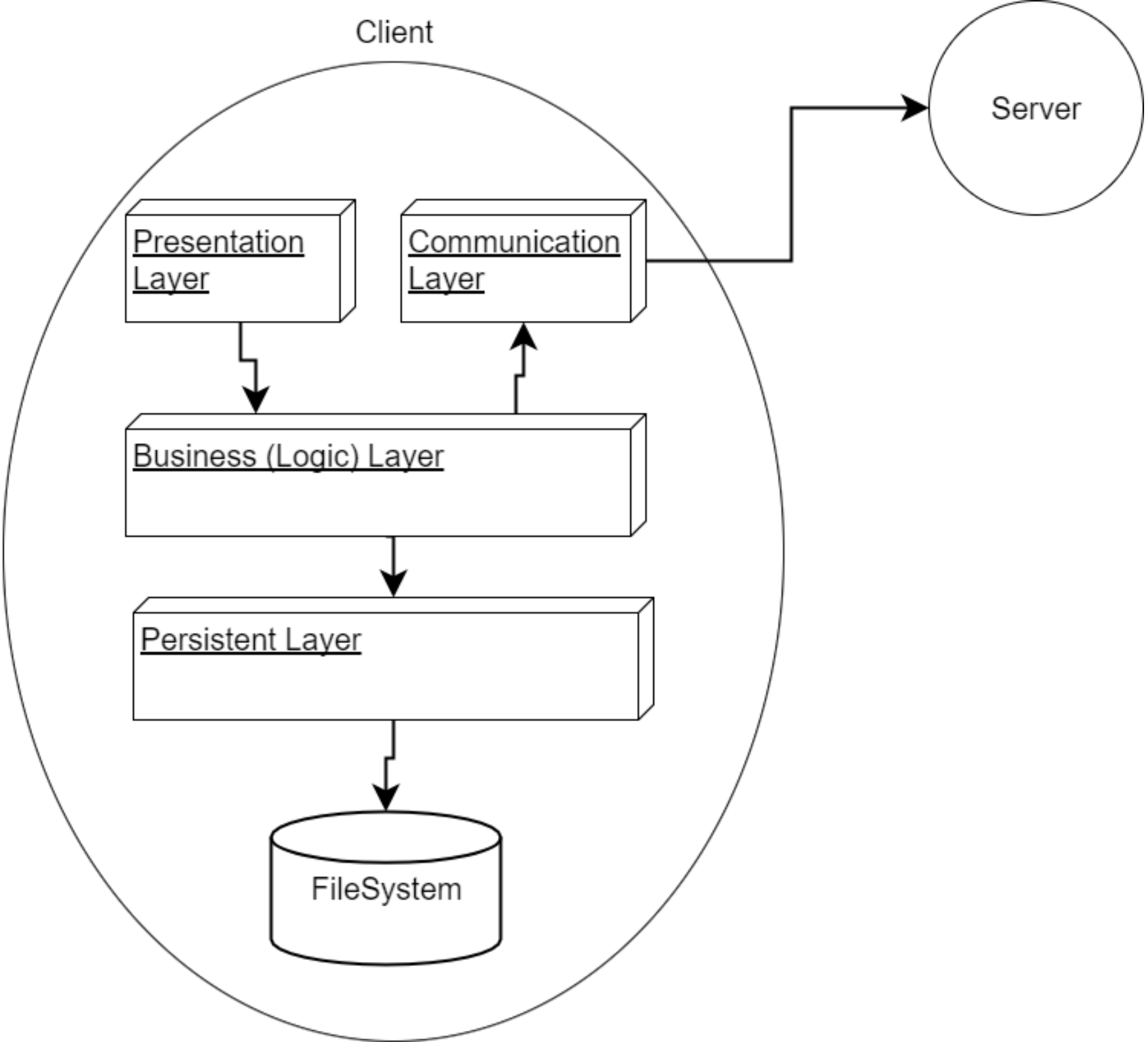
Methods

- Send(string url, string gourpID, string nickName, string messageContent) : IMessage
- GetTenMessages(string url): List<IMessage>

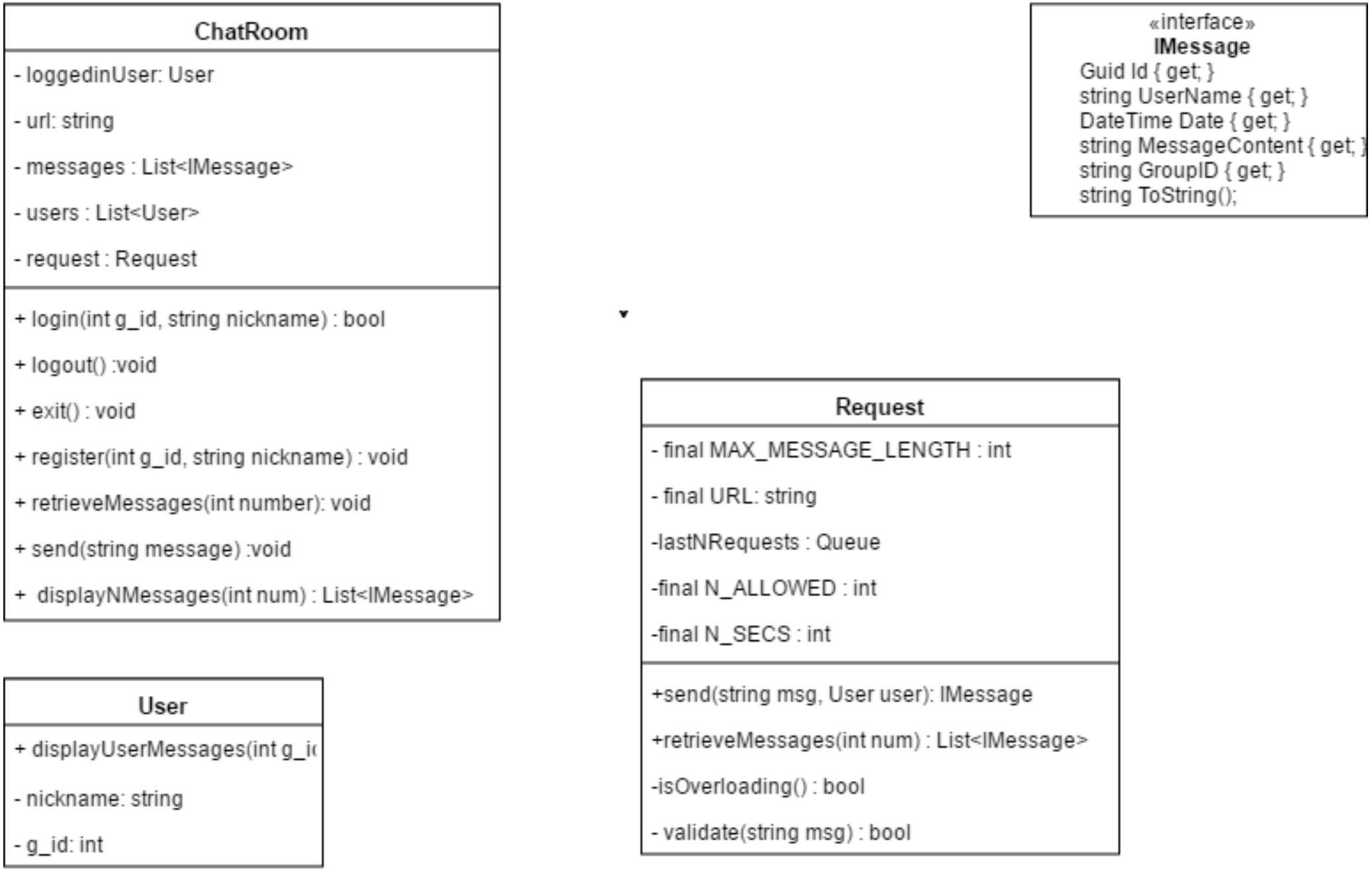
IMessage Interface

Methods

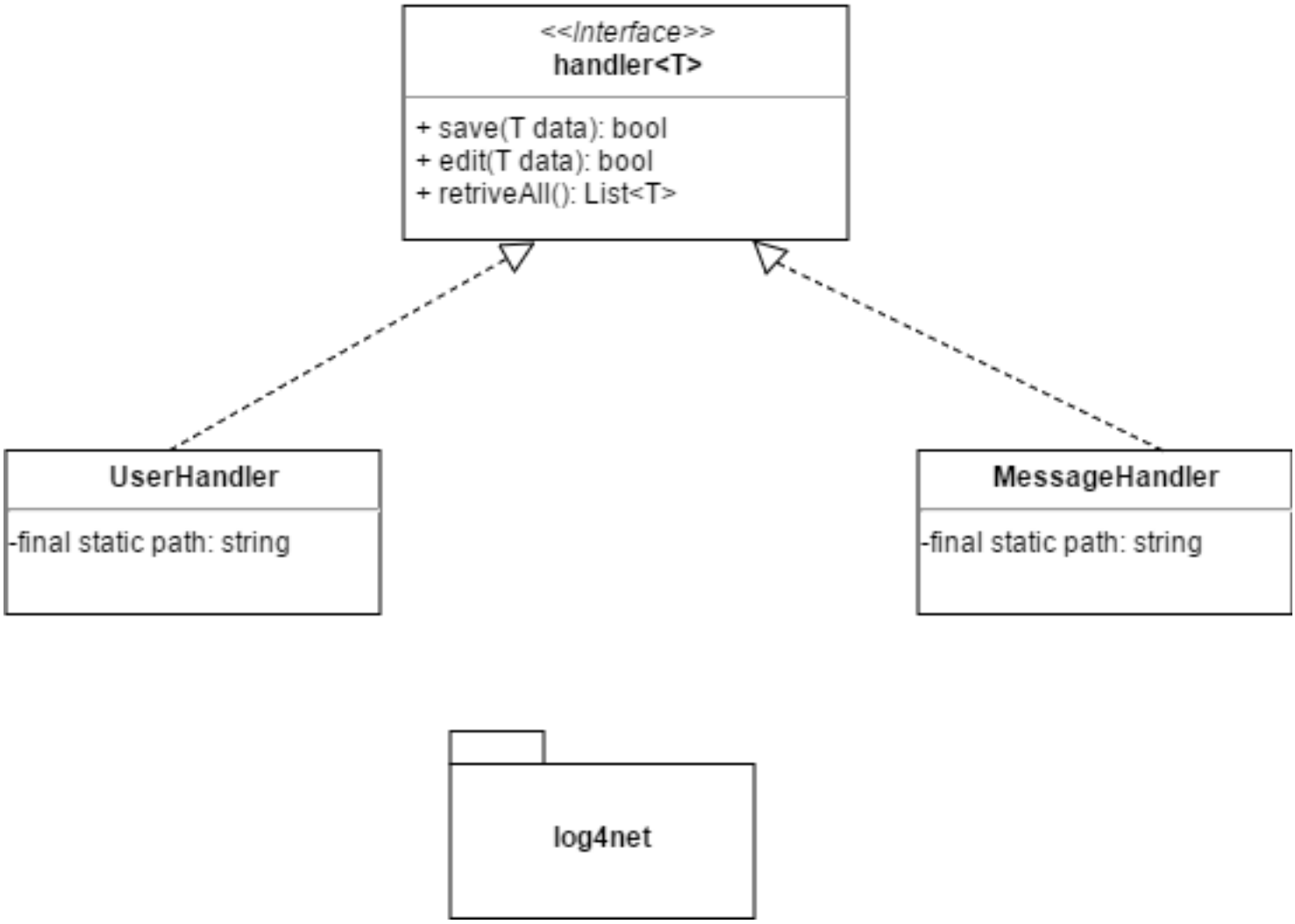
- Guid Id { get; }
- string UserName { get; }
- DateTime Date { get; }
- string MessageContent { get; }
- string GroupID { get; }
- string ToString();


Client
Server
Presentation Layer
Communication Layer
Business (Logic) Layer
Persistent Layer
FileSystem

Buisness Layer Diagram



Persistant Layer Diagram



Presintation Layer Diagram