## 1.1. Administrator/Server

### *Purpose of the server*

The administrator can access to the server, controlling the flow of protocols and the database, able to:

- o Change the database, although it requires a Database browser.
- o View the protocol messages in the console while the clients are interacting.
- o Shutdown directly the server through a command and prefix password.

```
[Server] [OUT] Replied : {"type":"RESPONSE","action":"LIST_APPOINTMENTS","requestId":"list
[Server] [IN] Received: {"type":"REQUEST","action":"LIST_MESSAGES","role":"DOCTOR","reques
[Server] [OUT] Replied : {"type":"RESPONSE","action":"LIST_MESSAGES","requestId":"list_mes
[Server] [IN] Received: {"type":"REQUEST","action":"LIST_MEASUREMENTS","role":"DOCTOR","re
[Server] [OUT] Replied : {"type":"RESPONSE","action":"LIST_MEASUREMENTS","requestId":"list
[Server] [IN] Received: {"type":"REQUEST","action":"GET_MEASUREMENT_VALUES","role":"DOCTOR
[Server] [OUT] Replied : {"type":"RESPONSE","action":"GET_MEASUREMENT_VALUES","requestId":
```

***Figure 1.*** *Protocol messages flow in console*



```
[Server] Connection closed for /
shutdown
Enter admin password to confirm shutdown:
[Admin] Shutting down server...
[Server] Shutting down thread pool and DB connection...
[DB] Connection closed.
-------------------------------------
[Server] Server stopped successfully.
```

***Figure 2.*** *Server shutdown through command and password*

### *How to make it work?*

1- Initialize the server by *run* in IntelliJ. Can be also done by running the jar. file.
2- The server will function automatically, meaning it doesn't require any action from the administrator to accept new clients.
3- The administrator can observe through the console the protocol messages, viewing who is connecting and if the client has closed the connection.
4- To shut down the server must write "shutdown" command in the console and password (it is stabilised as "admin").