

**Facultatea de Automatica si Calculatoare**

**Sectia Calculatoare**

**Chat System for Client Support**

**Documentation**

Zilai Denis

Grupa 30244; Anul 4, sem 1

Distributed Systems

Contents

[**1.** **Project objective** 3](#_Toc124209343)

[2. The analysis of the problem, modeling, scenario, use cases 3](#_Toc124209344)

[3. Design (design decisions, UML diagrams, class design, packages, User Interface) 3](#_Toc124209345)

[**4.** **README** 7](#_Toc124209346)

# **Project objective**

1. Requirements:

Develop a chat system to offer support for the clients of the energy platform if they have questions related with their energy consumption. The chat system should allow communication between the clients and the administrator of the system.

* 1. Functional requirements:

➢ The client application displays a chat box where clients can type messages.

➢ The message is sent asynchronously to the administrator, that receives the message together with the client identifier, being able to start a chat with the client.

➢ Messages can be sent back and forth between the client and the administrator during chat session.

➢ The administrator can chat with multiple clients at once.

➢ A notification is displayed for the user when the other user reads the message.

➢ A notification is displayed for the user (e.g., typing) while the user from the other end of communication types its message.

# The analysis of the problem, modeling, scenario, use cases

Functional requirements in this project:

* The client has a text message box where messages can be typed and sent
* The message is sent asynchronously with the name of the person that sent the message

# Design (design decisions, UML diagrams, class design, packages, User Interface)

The system was implemented using Web Sockets technology and Java Spring. There have been 4 classes implemented in this project:

* Main class: the class that runs the spring application
* ChatController class: represents basically the server part of the application
* Message class: this class contains the object used for the encapsulation of the sent and received message
* WebConfig class: represents the broker (the part that is in charge of transmitting the messages

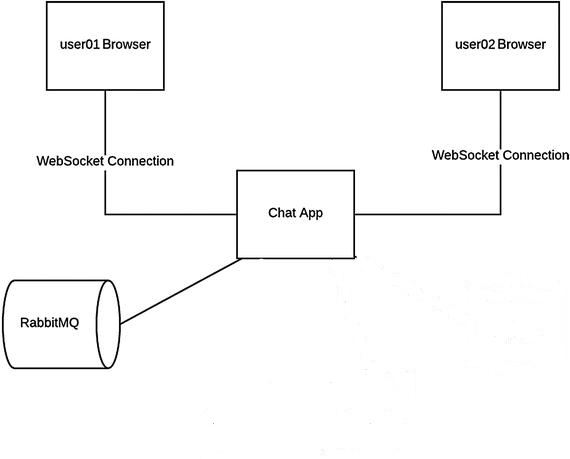
For the frontend part of the application, the technologies used are: HTML, CSS for a simple interface, and JavaScript for the functionality.

An important functionality of the application is the possibility of having multiple users using the application at the same time.

Deployment:

Diagram

Description automatically generatedArchitecture:



# **README**

To run this application, start the Assignment3Application class main, then open a browser and access the localhost:8080 to be logged in as a client. To chat with another user, open another browser window with the same URL.