INF
Informatik

Hochschule Reutlingen
Reutlingen University

Media and Communication Informatics

Cloud Computing WS 2017

Prof. Dr. Marcus Schöller

Submitted on

2017/10/25

Authors

Alexandros Konstantakos, 741590

Zahid Ibnu Yusuf, 741463

1 Requirements

These requirements were defined in the exercise. We split them into functional and non-functional, and also defined a couple of our own marked in blue, of course with a lower priority.

1.1 Functional

1. A new user has to register with the website first by providing a chat name.
2. A marked message is generated and sent to the chat room each time a new user connects to the chat room or leaves the chat room.
3. A user can request a list of all online users using the "\list" command; that list is only sent to the requestor.
4. A user can send a message to one dedicated recipient (private message) instead of to all online users in the chat room.
5. A user can send multi-media files to the participants of a chat room or via a private message, e.g., pictures, movies, sound files.
6. A user can press on a button to get information about the available commands.

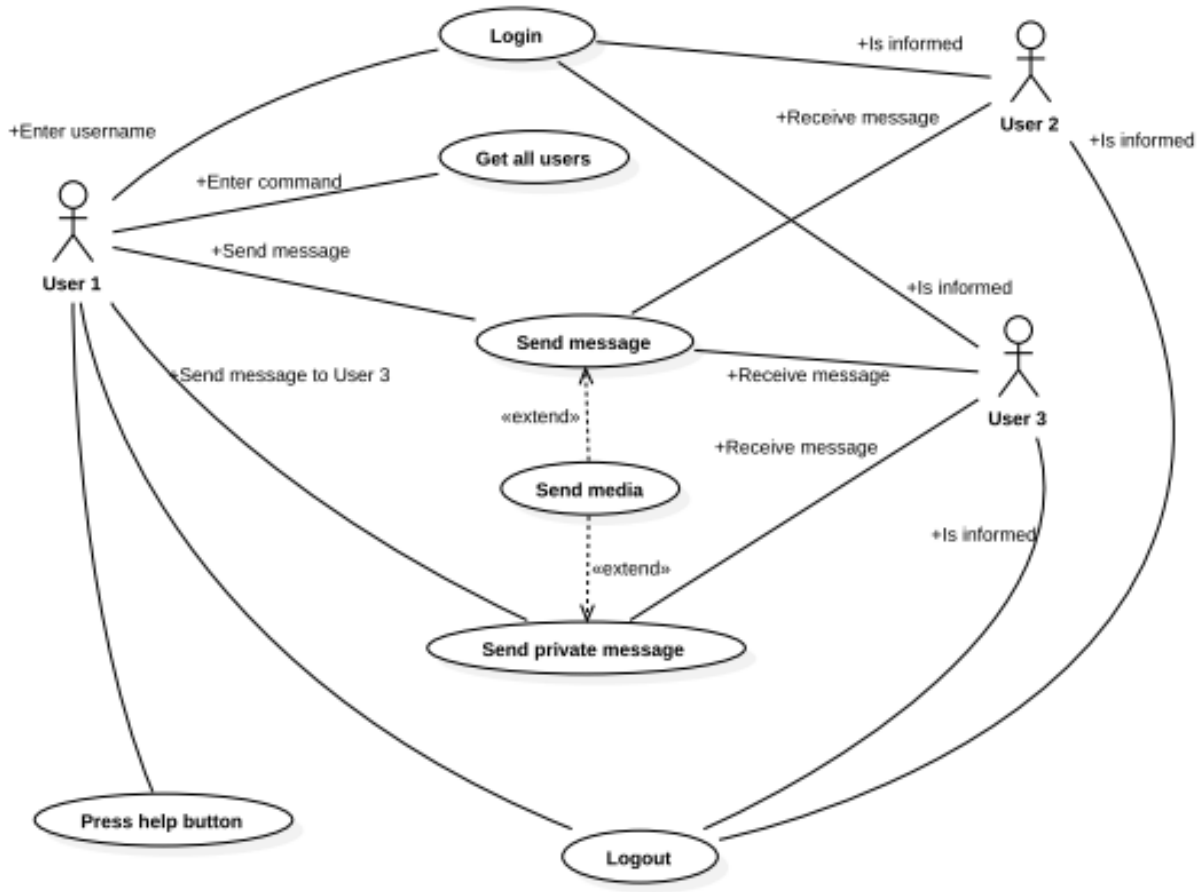
1.2 Non-Functional

1. In the chat room, all messages should be displayed in chronological order with a timestamp and the chat name of the user who has posted the message.
2. It should be a single page web application, to avoid reloading the page.

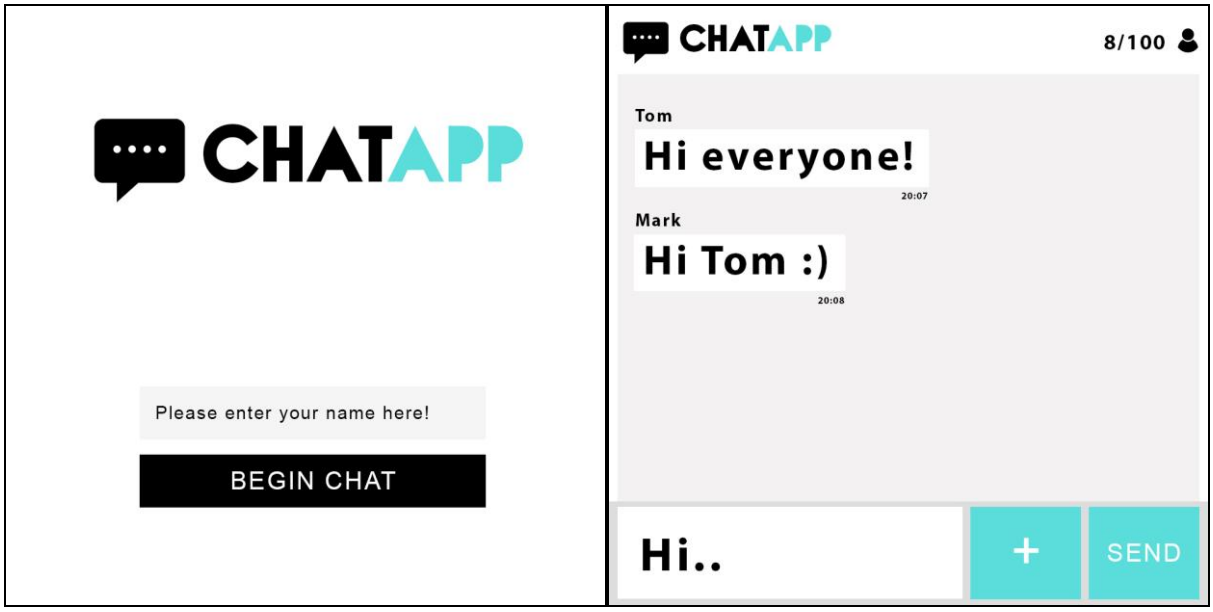
2 Design

We started our design of the application by creating a use case diagram and some mockups based on the requirements. After that we proceeded with the creation of a simple component diagram. Finally, to minimize merge conflicts and get a good workflow, we created a class diagram and splitted the source code into multiple files.

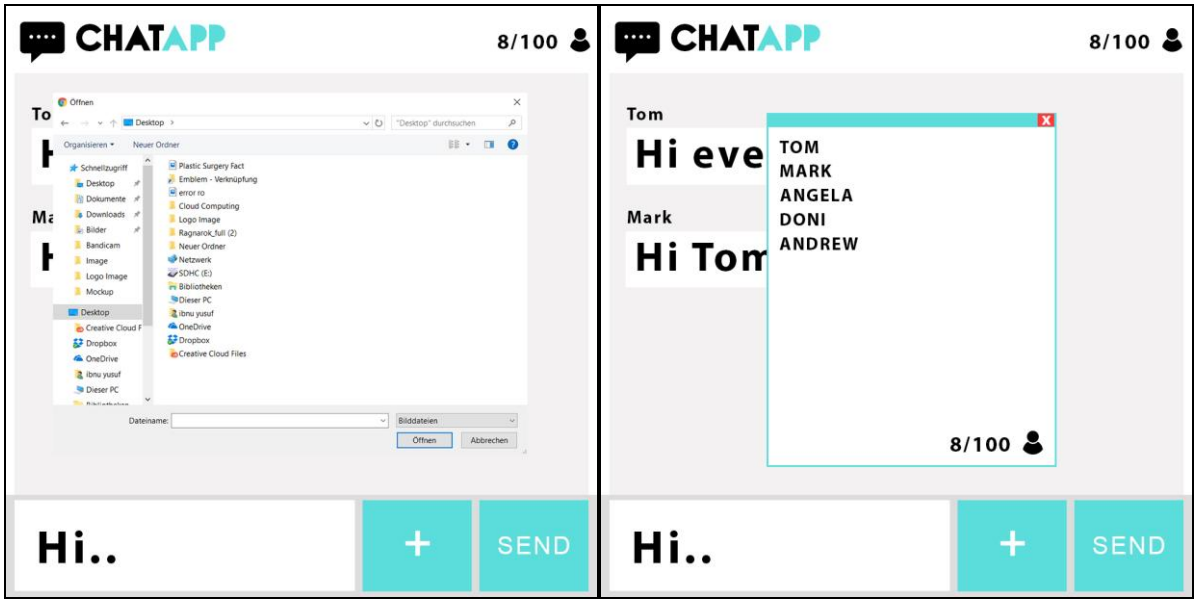
2.1 Use Case Diagram



2.2 Mockups

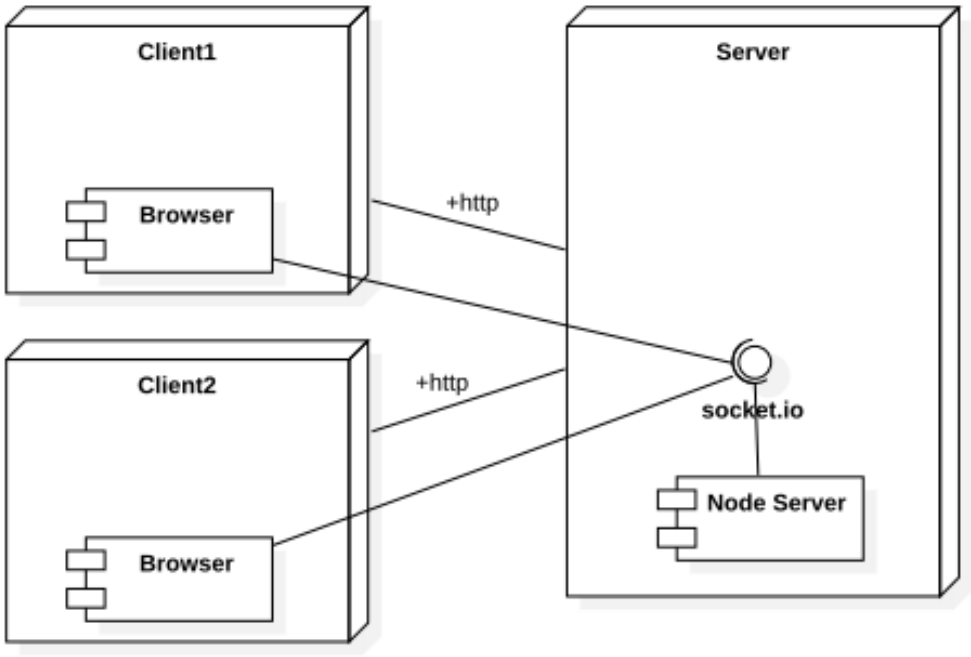


The login screen *The chat*

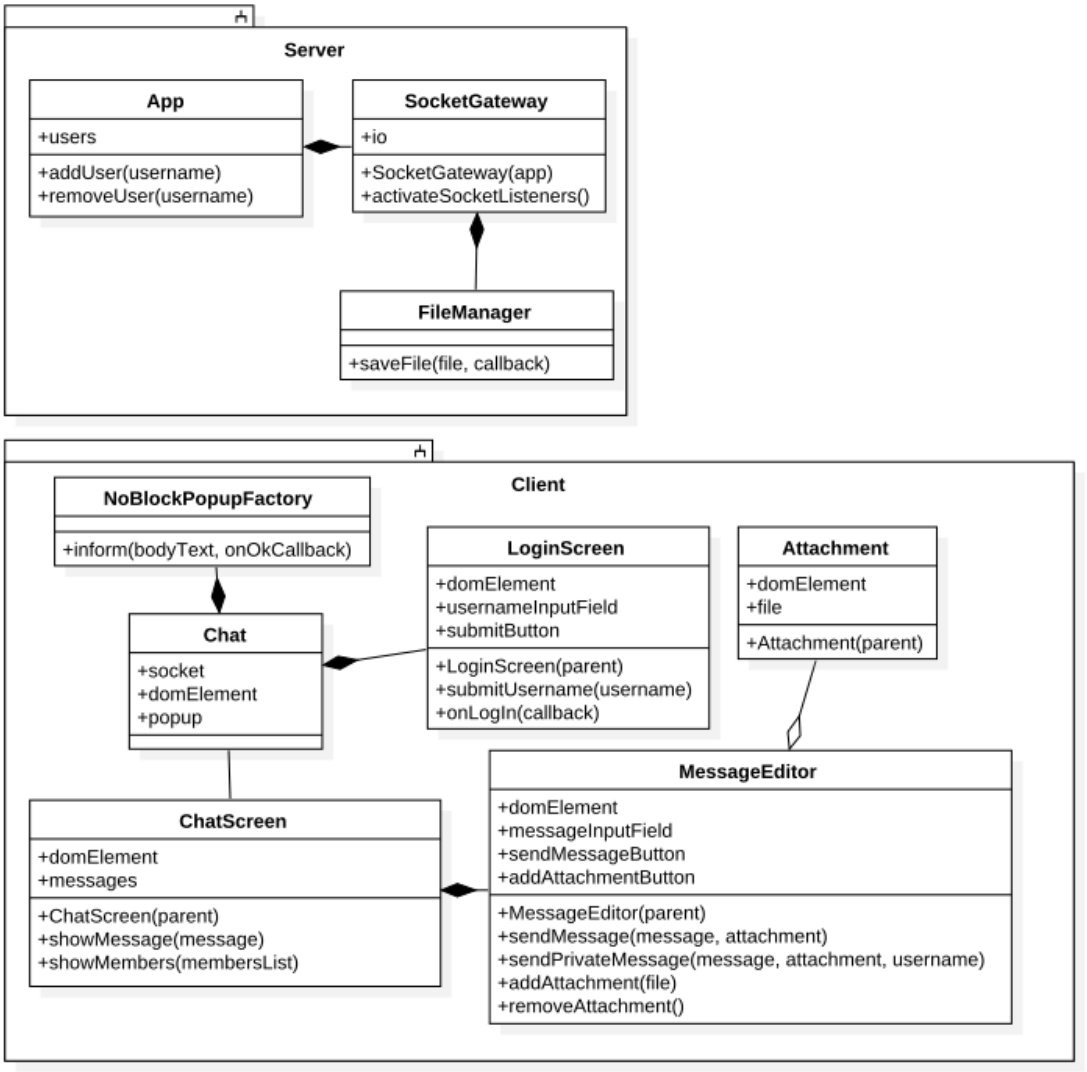


Uploading media *A list of the users*

2.3 Component Diagram



2.4 Class Diagram



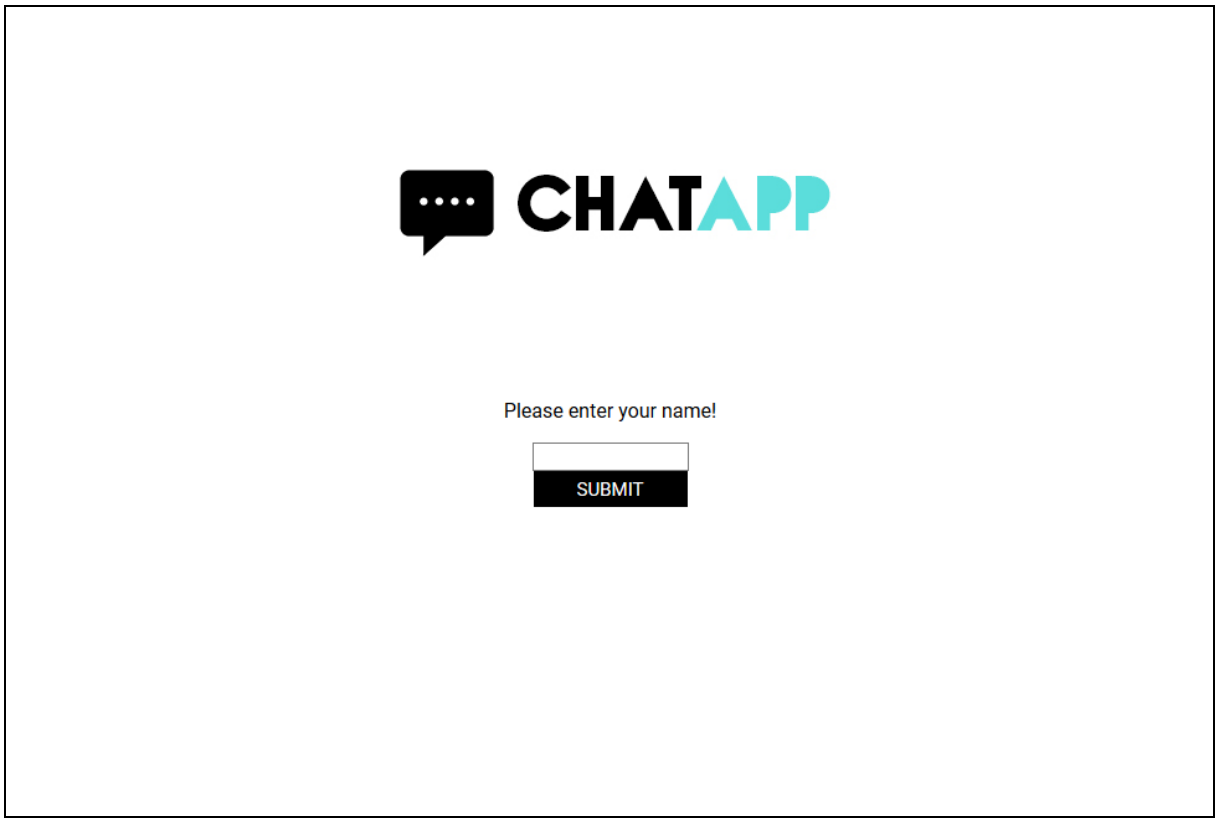
3 Results and Lesson Learned

After managing to fullfil all the requirements, in this chapter we present some instructions to get started, and the product itself in pictures.

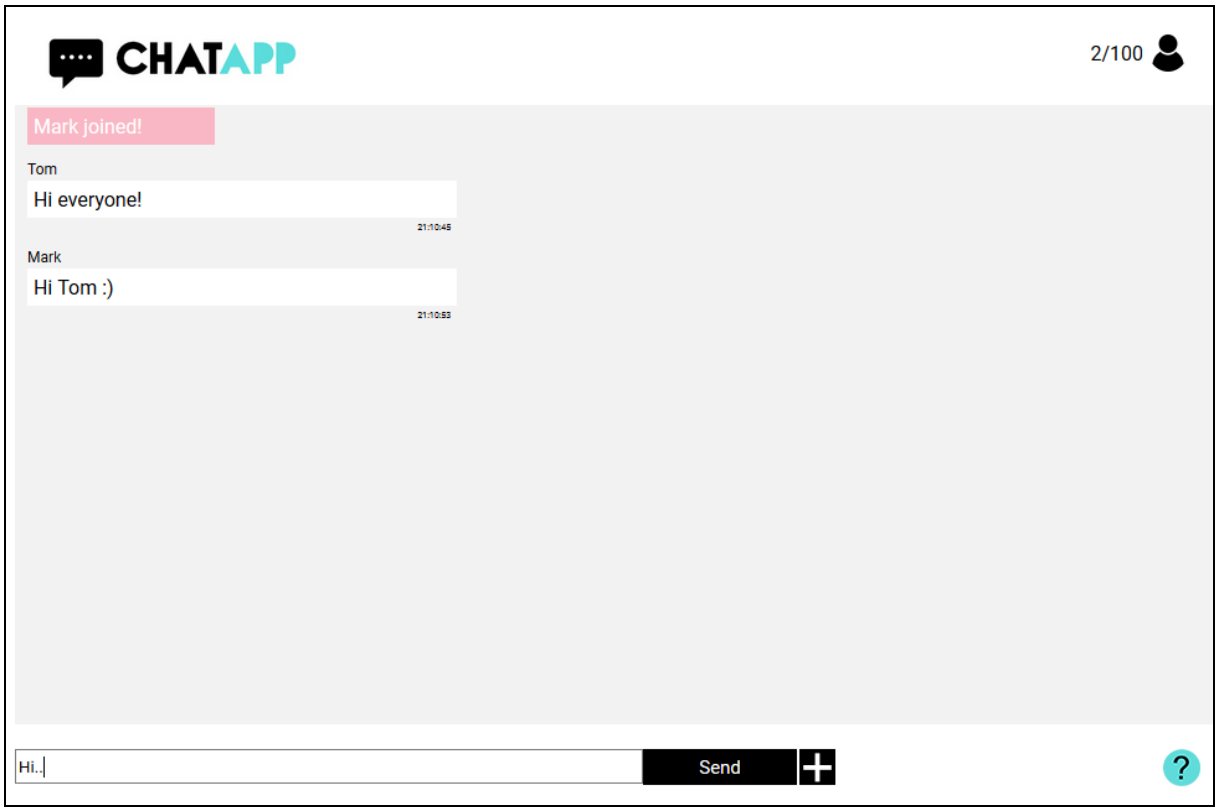
3.1 Installation and running the server

1. Go into the src folder and open a cli window
2. Enter npm install. This installs all the necessary libraries defined under package.json
3. Enter npm start. This starts the server as defined under package.json
4. Now your server is running on port 3001
5. Open a web browser and enter localhost:3001 in the address field

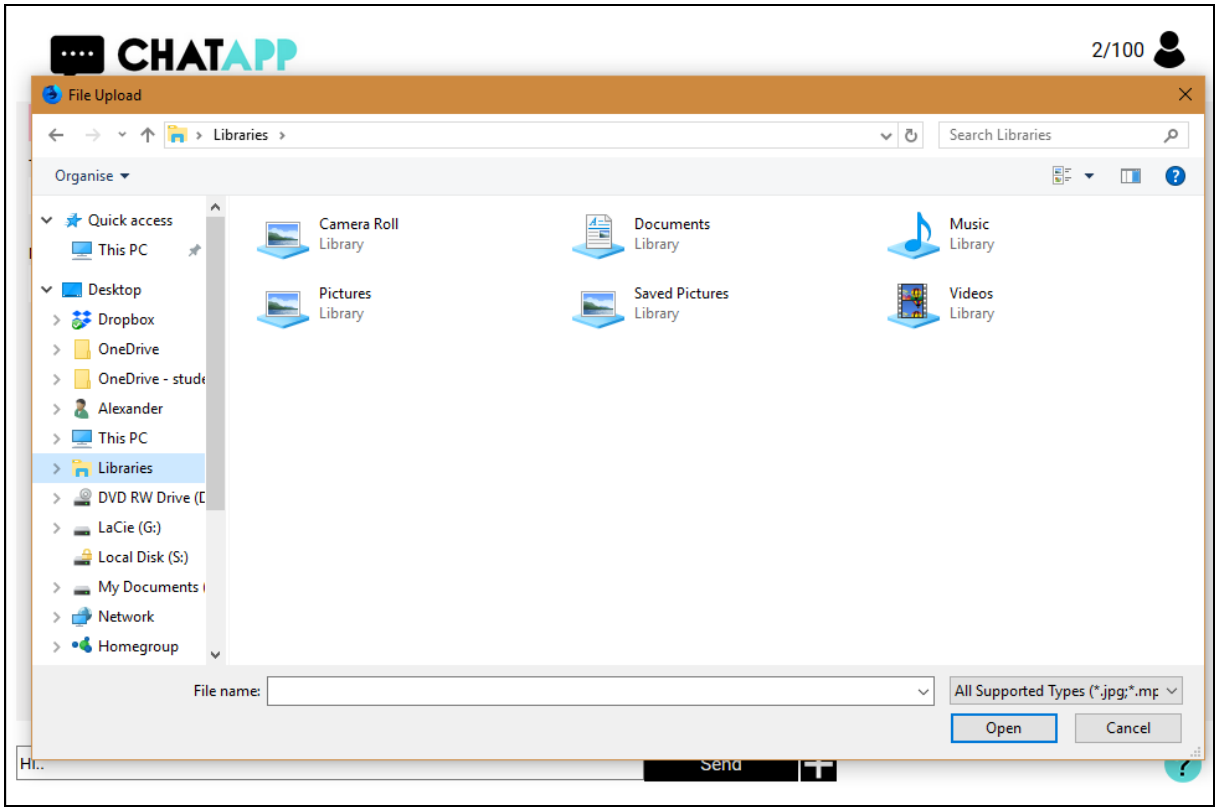
3.2 Pictures of the product



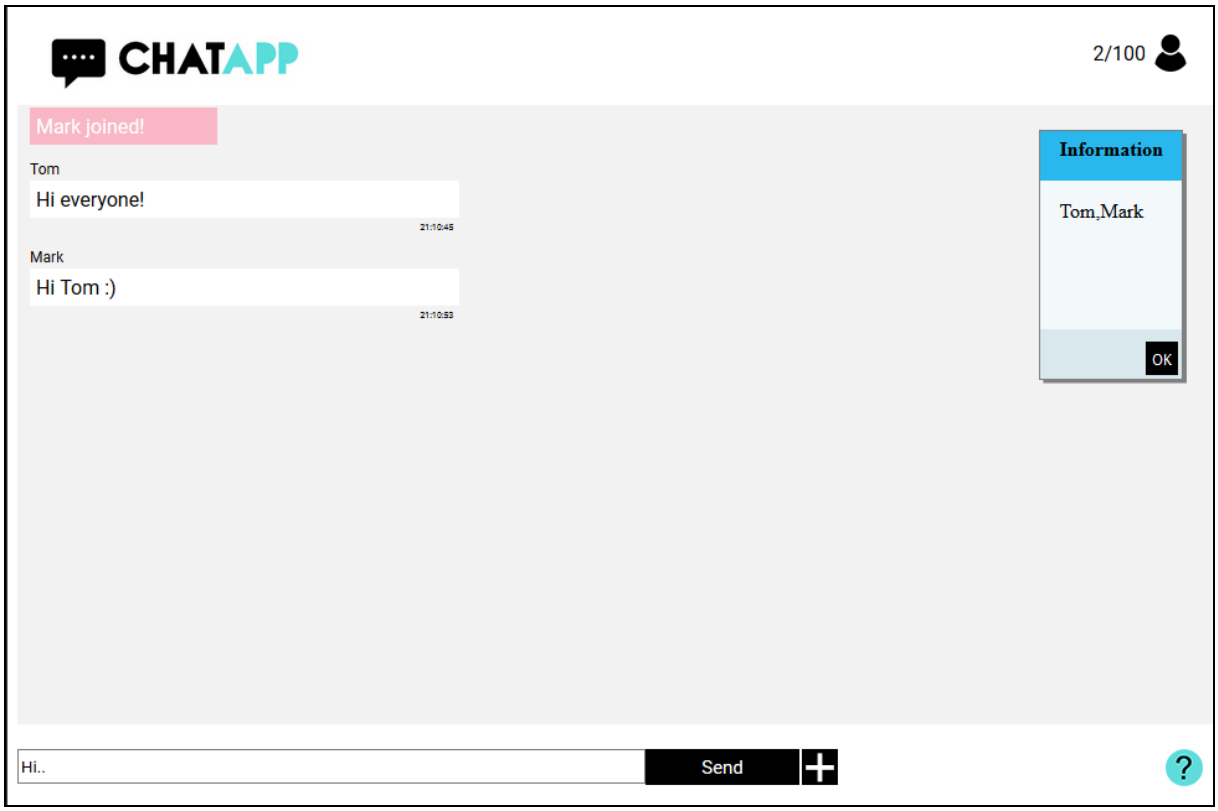
The login screen



The chat



The file chooser in the chat



Getting a list of the users

3.3 Future improvements

A suggested future improvement would be to implement a build mechanism that combines automatically all the .js files into one. This way the client needs to make less GET requests to the Server, improving the loading time.

3.4 Lesson learned

Changing a webpage using JavaScript functions without loading a new HTML document, gives a responsive feeling to the user.