UDC 004.4;004.6

SYSTEM OF COMMUNICATION THROUGH THE INTERNET (WEB-CHAT)

Oleksandr Hribanov, Marta Mashevska

Lviv Polytechnic National University
12, Bandera St., Lviv, 79013, Ukraine

This article reveals the problems of creating a system of communication between people, and also analyzes different approaches to communication between people, namely: forums, communication with people via e-mail and messengers. In the practical part, communication system is created on the basis of the messenger using the latest technologies. The scientific novelty of the obtained results is that with the use of modern technologies of a web application for messaging between users has been developed, which is characterized by mobility, simplicity and ease of use. An approach is presented for developing a web application for messaging using a set of universal software modules and components, or those that are easily modified according to the needs of a particular subject area, which allows you to quickly and efficiently develop software packages in particular fields.

Keywords: *online communication, Socket.IO technology, MPA and SPA architectures.*

Introduction. Today is a period of rapid growth of information technology. At this stage, the Internet is the main generator of global trends. Moreover, with the development of new technologies, he moved to mobile phones. This allowed people to stay online almost all the time.

The ability to easily and seamlessly communicate on the Internet - one of the most distinctive features of today. Thanks to the World Wide Web, today you can find many like-minded people and even friends not only somewhere nearby, but also in the farthest corners of the planet, receive a variety of information from them and respond to it instantly. This can be done through chats - software tools that are designed specifically for real-time communication. They blur the boundaries between people, make communication more accessible and convenient. Chats are found on commercial sites, in the form of technical support, video chats, television chats. They work great and have their own audience.

The current state of the problem. There are now many ways to communicate online. The most popular of them:

1. Forums - the first way to communicate online, there were forums, it's usually a simple website, with a minimum of features that allow you to correspond, throw funny pictures to each other and more. The main advantage of the forums was that it was the first opportunity to communicate at a distance thanks to a new, at that time, technology - the Internet. The main drawback was that the functionality could not fully replace live communication, as now, messengers do.

2. Messengers - this is a fairly modern communication technology. The advent of this technology has made it possible to communicate wherever you are, because messengers are usually cross-platform, which allows you to be online always, whether from a phone, computer, tablet, etc. In just a few steps, people can transfer any data to each other. The only problem is that not all people still have smartphones, which does not allow them to communicate with other people always and everywhere. Despite this, the messengers gave you a great opportunity to always communicate with the people you need.

3. The least popular way to communicate now is to communicate via email. Since there are now so many messengers, applications for communication, e-mail went a long way because it was not competitive. The benefits of communication via email, I cannot even name, because I do not see the point of communication like this. Disadvantages can be listed for a long time, but let's focus on the main ones: the ability to communicate with only one person; inability to send large files.

Each architecture has its advantages and disadvantages, and the MPA and SPA architectures are no exception [1, 2]. It is important to know these pros and cons to be able to determine what is best for your idea.

In the case of MPA, we know that the server side takes full responsibility for all the functions of generating the view, while the client part deals exclusively with the visualization of the generated static content. It is worth noting that in this case, the implementation of a "fat" client with its business logic and maintaining a complex state is difficult or impossible.

In the case of SPA, the server does not generate a ready-made view, but instead during the program exchanges information packets with the client. In this case, the client may have its own level of business logic processing, moreover, the client may store on its side some data that will depend on the state of the client. In this case, we can say that the client can be implemented as a "fat" client.

Like any other architecture, SPA has certain disadvantages and advantages over others.

Disadvantages of SPA regarding MPA:

- Client is a rather difficult interface program that needs to be downloaded for each client. However, the availability of resource caching by modern web browsers minimizes the negative consequences of this shortcoming;
- SEO - SPA complicates site optimization for search engines. Because most web pages are built on the client side, search engine bots do not see this page as a user.
- Advantages of SPA over MPA:
- higher program speed and download
- high level of interactivity of the user interface, as well as improved perception of the user interface, as data is downloaded from the server in the background;
- division of the program into client (interface, external representation) and server (background) part, which allows to implement many different external representations for different platforms, including mobile, interacting on one API with the server part of the program;

- the ability to implement a "fat" client, which will provide greater dynamics on the client side.

The goal of the work. The purpose of this work is to develop a web application that allows users to comfortably communicate with each other.

Briefly about the main Socket.IO library. Socket.IO is a library that provides real-time, two-way, and event-based communication between the browser and the server [3]. Consists of:

- Node.js server
- Javascript client library for the browser (which can also be run from Node.js) [4, 5].
- Main features:
- Reliability - Connections are established even if there are proxy servers and load balancers, a personal firewall, and antivirus software. To that end, he relies on Engine.IO, which first installs a lengthy survey and then tries to switch to the best vehicles that are "tested" from the side, like WebSocket.
- Support for automatic recovery - unless otherwise specified, the disconnected client will try to connect again until the server reappears.
- Shutdown detection - the heartbeat mechanism is implemented at the Engine. IO level, which allows both the server and the client to know when the other is no longer responding. This functionality is achieved through timers installed on both the server and the client, with timeout values (pingInterval and pingTimeout parameters) being shared during the connection. These timers require that any subsequent client calls be routed to the same server.
- Binary support - you can release any serialized data structures, including: ArrayBuffer and Blob in the browser and ArrayBuffer and Buffer in Node.js.
- Multiplexing support - to create a partition of problems in the program (for example, module or permissions), Socket.IO allows you to create multiple namespaces that will act as separate communication channels, but will have the same basic connection [3].
- Number support - You can define arbitrary channels in each namespace, called Rooms, where sockets can join and leave. You can then broadcast to any room, reaching each outlet that is connected to it. This is a useful feature for sending notifications to a group of users or, for example, to a specific user connected on multiple devices.

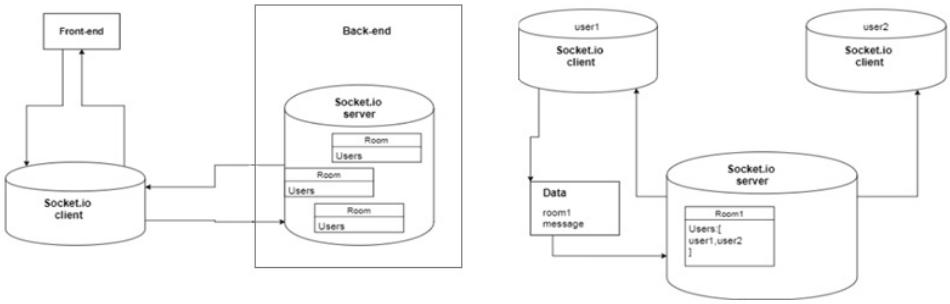

Fig. 1. Messaging diagram

Data is transferred from the frontend to the socket (client). After that, the same data is distributed on the server [6]. The data sent by users has a room where the message arrives. The server finds this room and sends messages to the users who are in this room. The server sends data to the client socket, and they in turn - to the frontend (Fig.1).

Briefly about the web application. The general view diagram of the web application clearly shows all the connected elements and their interaction with each other. As you can see, the frontend interacts with its Redux repository [7], with the Auth0 Service and with the backend [8].

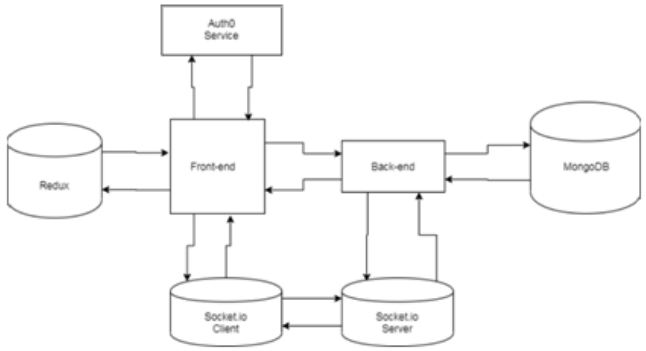

Fig. 2. Diagram of the general view of the web application

Practical implementation. Usually a visitor to the site within 10-15 seconds should understand how to start using the site. Because ease of use and a clear interface were among the basic requirements, we created a software product that captivates the user at first sight.

You can get acquainted with the developed site by the link: https://practick.herokuapp.com/

So, the home page is greeted with animation. Of course, you should start working in the web application by registering or logging in. To do this, the user must select the desired option: SIGN IN or SIGN UP (Fig. 3). There are two ways to sign up: Google services or email. In the second case, the user also needs to come up with a password (Fig. 4).

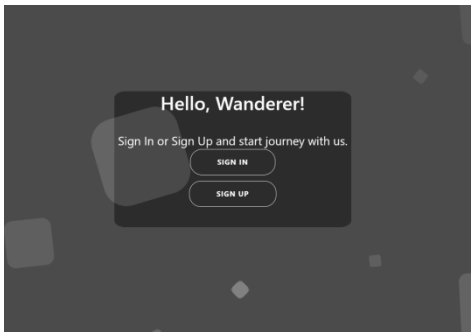

Fig. 3. Home page

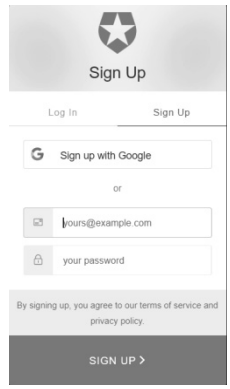

Fig. 4. Registration in the web application

There are certain requirements for the password in order to make it strong, namely the password must consist of at least 8 characters, there must be uppercase and lowercase letters, numbers from 0 to 9, as well as special characters (! @ &, Etc.) (Fig. 5).After registration, the user is asked to confirm the account via e-mail and log in to the web application (Fig. 6).

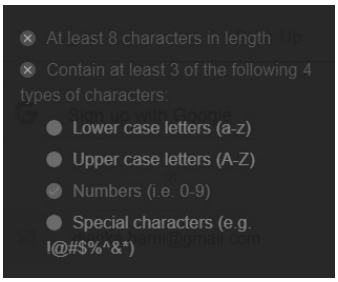

Fig. 5. Requirements for the password

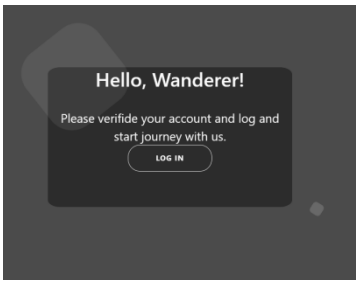

Fig. 6. Login page

On the main page, the user selects the person with whom he wants to communicate (Fig. 7).After that, users start their conversation (Fig. 8).

Fig. 7. Select a chat on the home page

Fig. 8. Chat Page

Conclusions. As a conclusion, it is worth to say that technologies are developing lightning fast and we all want to be up-to-date and use the newest and most convenient technologies. Web applications have clear advantages over their software counterparts. This is their mobility, ease of creation and use, convenience. That is why this technology is rapidly gaining popularity, both among users and developers. According to the practical part, it can be concluded that having a certain set of universal software modules and components, or those that easily change to meet the needs of a particular subject area, you can quickly and efficiently develop software packages in a particular field.

REFERENCES

1. Forum on multi-page and one-page applications (MPA, SPA) [Online]. – https://dou.ua/forums/topic/25444/ (in English)
2. Information about web applications [Online]. – https://www.centum-d.com/uk/veb-dodatok-yogo-harakteristiki (in English)
3. Socket.IO documentation [Online]. – https://socket.io/docs/ (in English)

4. Jeanine Meyer (2018). “HTML5 and JavaScript Projects: Build on your Basic Knowledge of HTML5 and JavaScript to Create”, 501 p. (in English)
5. Joe Morgan (2018). “Simplifying JavaScript: Writing Modern JavaScript with ES5, ES6, and Beyond”, 279p. (in English)
6. David Hows, EelcoPlugge, Peter Membrey, Tim Hawkins (2013). “The Definitive Guide to MongoDB A Complete Guide to Dealing with Big Data using MongoDB”, 319 p. (in English)
7. A short guide to Redux for beginners [Online]. – https://tproger.ru/translations/redux-for-beginners/ (in English)
8. Information about Auth0 [Online]. – https://webformyself.com/bezopasnaya-autentifikaciya-s-auth0/ (in English)

DOI 10.32403/2411-9210-2020-2-44-106-111

СИСТЕМА СПІЛКУВАННЯ ЧЕРЕЗ ІНТЕРНЕТ (ВЕБ-ЧАТ)

О.В. Грібанов, М.В. Машевська

Національний університет «Львівська політехніка»
вул. С. Бандери, 12, м. Львів, Україна, 79013
o.gribanov2000@gmail.com, marta.v.mashevska@lpnu.ua

Методологія ґрунтується на специфікації вимог до програмного забезпечення, підходах до проектування веб-орієнтованих додатків, аналізі аналогічних систем для спілкування та обміну повідомленнями між користувачами в мережі Інтернет, огляді та виборі оптимальних технологій для реалізації поставлених завдань.

У статті виконано аналіз та обґрунтований вибір підходів до розроблення веб-додатку, який дозволяє користувачам зручно та ефективно спілкуватися між собою в мережі. Розглянуто різні способи комунікації, а саме: форуми, спілкування за допомогою електронної пошти та месенджери.

Наукова новизна одержаних результатів полягає в тому, що з використанням сучасних технологій розроблено веб-додаток для обміну повідомленнями між користувачами, який характеризується мобільністю, простотою та зручністю використання.

Представлено підхід щодо розроблення веб-додатку для обміну повідомленнями за допомогою набору універсальних програмних модулів та компонентів, або тих, які легко змінюються відповідно до потреб певної предметної області, що дозволяє швидко та ефективно розробляти програмні пакети в певній галузі.

Ключові слова: *мережа interner, веб-додаток, способи комунікації, програмний модуль.*

Стаття надійшла до редакції 25.10.2020
Received 25.10.2020