МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИИ

федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский университет информационных технологий механики и оптики

Мегафакультет трансляционных информационных технологий

Факультет информационных технологий и программирования

Лабораторная работа №7

По дисциплине «Web-программирование»

Добавление обработки данных реального времени с использованием Web-Socket канала

Выполнил студент группы
***M33122*:**
Федотенко Николай Владимирович

Проверил:
Приискалов Роман Андреевич

САНКТ-ПЕТЕРБУРГ

2022

Цель работы:

Добавление виджета, отображающего последние изменения данных модели по выбору.

Краткие теоретические сведения:

- **WebSocket** – это протокол связи поверх TCP-соединения, предназначенный для обмена сообщениями между браузером и веб-сервером в режиме реального времени.
- **Gateway** (в NestJS) – это класс, аннотированный декоратором ***@WebSocketGateway()***. Технически, gateways (*шлюзы*) платформенно независимы, что позволяет им быть совместимыми с любой WebSocket библиотекой при создании адаптера, в частности, с библиотекой ***socket.io***.

Ход выполнения работы:

Данная лабораторная работа выполнена в операционной системе *macOS*.

Для выполнения был добавлен канал для работы с соединениями по протоколу WebSocket используя средства библиотеки *Socket.IO*.

Используемая IDE: *WebStorm 2022.1* (by JetBrains)

1. Установка необходимых компонентов:

```
npm i --save @nestjs/websockets @nestjs/platform-socket.io @types/socket.io
```

2. Создание отдельного модуля под gateway:

```
nest g module chat
```

3. Реализация класса ChatGateway (*src/chat/chat.gateway.ts*):

```
import {
  MessageBody,
  SubscribeMessage,
  WebSocketGateway,
  WebSocketServer,
} from '@nestjs/websockets';

@WebSocketGateway({ namespace: 'chat' })
export class ChatGateway {
  @WebSocketServer()
  server;

  @SubscribeMessage('message')
  handleMessage(@MessageBody() message: string) {
    console.log('handled message:', message);
    this.server.emit('message', message);
  }
}
```

4. Реализация контроллера чата (*src/chat/chat.controller.ts*):

```
import { ApiOperation, ApiTags } from '@nestjs/swagger';
import { Controller, Get, Render } from '@nestjs/common';

@ApiTags('Chat')
@Controller()
export class ChatController {
  @ApiOperation({
    summary: 'Enter chat',
    description: 'Cool anonymous chat on my page!',
  })
  @Get('chat')
  @Render('chat')
  showChat() {
    return;
  }
}
```

5. Реализация клиентской части чата:

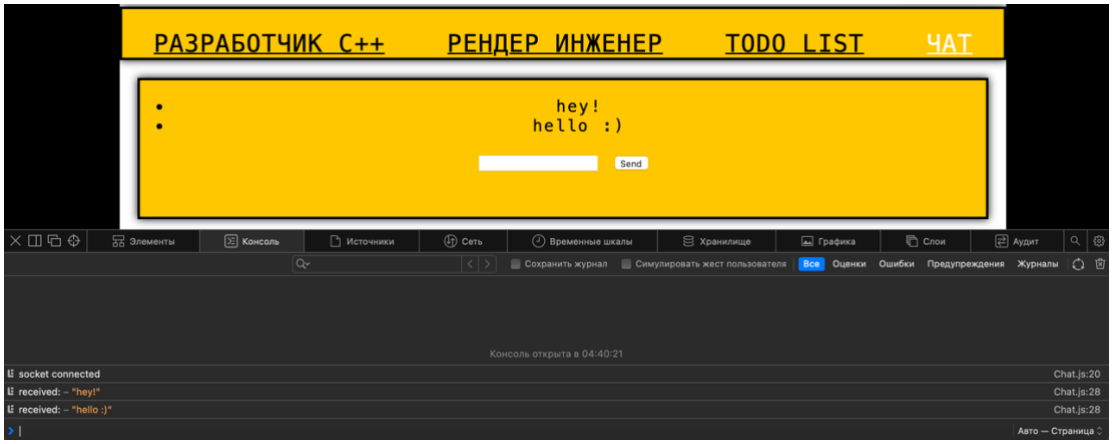
public/JS/Chat.js:

```
const message = document.getElementById('message');
const messages = document.getElementById('messages');

message.addEventListener('keypress', (event) => {
  if (event.key === 'Enter') {
    event.preventDefault();
    send();
  }
});

const chatSocket = io('/chat');

function send() {
  chatSocket.emit('message', message.value);
  message.value = '';
  message.focus();
}

chatSocket.on('connect', () => {
  console.log('socket connected');
});

chatSocket.on('disconnect', () => {
  console.log('socket disconnected');
});

chatSocket.on('message', (message) => {
  console.log('received:', message);
  receiveMessage(message);
});

function receiveMessage(message) {
  messages.appendChild(createMessage(message));
}

function createMessage(message) {
  const li = document.createElement('li');
  li.appendChild(document.createTextNode(message));
  return li;
}
```

views/chat.hbs:

```
<!DOCTYPE html>
<html lang="ru">
<head>
  {{> head}}
</head>
<body>
<div class="card">
  <!-- Header -->
  <header>
    {{> header }}
  </header>

  <main>

    <div class="inf-block" id="contact">
      <div>
        <ul id="messages"></ul>
      </div>

      <div>
        <input id="message" type="text" spellcheck="false" />
        <button onclick="send()">Send</button>
      </div>
    </div>

  </main>
  <!-- Footer -->
  <footer>
    {{> footer }}
  </footer>
</div>
</body>
<script src="https://cdn.socket.io/4.5.0/socket.io.min.js"
integrity="sha384-
7EyYLQZgWBi67fBtVxw60/OWl1kjsfrPFcaU0pp0nAh+i8FD068QogUvg85Ewy1k"
crossorigin="anonymous"></script>
<script src="JS/Chat.js"></script>
</html>
```

6. Проверка результата:



Вывод:

Я добавил виджет, отображающий отправляемые сообщения пользователями по протоколу WebSocket (*анонимный чат*), используя средства библиотеки *Socket.IO*.