

WEB. Работа с NodeJS.

Урок 12. Практика. Кликер.

Задача.

Вам дается готовый клиентский код.

Вложение будет содержать 2 файла HTML и изображение.

Ваша задача в том, чтобы написать сервер, отвечающий на 4 запроса.

Итак, файлы:

1 - auth.html

Файл с аутентификацией клиента, где расположена форма для входа с логином и паролем.

Клиент вводит данные, сервер проверяет правильность логина и пароля и возвращает содержимое новой страницы index.html

Код:

```
fetch('http://localhost:3000/login', {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify({
    username: username,
    password: password
  })
})
.then(response => {
  if (response.ok) {
    return response.text(); // Получаем HTML в виде текста
  } else {
    throw new Error('Ошибка входа');
  }
})
.then(html => {
  window.document.write(html);
})
```

```
    })  
    .catch((error) => {  
        alert('Ошибка входа');  
    });
```

В случае успешной обработки, ваша страница перерисовывается на новый HTML благодаря конструкции `window.document.write(html)`;

2 - index.html

Здесь веб приложение - кликер.

Логика здесь простая, клиент кликает по кнопке, зарабатывая очки. В это время, на каждый 10 клик отправляется POST запрос для актуализации данных о пользователе.

Каждый 10й клик вызывается следующий метод:

```
fetch('http://localhost:3000/login', {  
    method: 'POST',  
    headers: {  
        'Content-Type': 'application/json'  
    },  
    body: JSON.stringify({  
        username: username,  
        password: password  
    })  
})  
    .then(response => {  
        if (response.ok) {  
            return response.text(); // Получаем HTML в виде текста  
        } else {  
            throw new Error('Ошибка входа');  
        }  
    })  
    .then(html => {  
        window.document.write(html);  
    })  
    .catch((error) => {  
        alert('Ошибка входа');  
    });
```

Каждый 100й клик появляется предложение об увеличении набираемых очков:



И при нажатии на этот блок, на сервер отправляется PUT-запрос об обновлении данных пользователя, где его `additionalCoins` должны быть увеличены.

```
if (tapCounter % 100 == 0) {  
    createEncreaser();  
}
```

В методе `createEncreaser()`; вы найдете все данные, отправляемые клиенту.

Так же, не забывайте о картинке, которая возвращается сервером:

```
.wave-button {  
    position: relative;  
    width: 200px;  
    height: 200px;  
  
    background-color: #007bff;  
    background-image: url(http://localhost:3000/dollar-symbol.png);  
    background-size: cover;  
    border: none;  
    border-radius: 50%;  
    cursor: pointer;  
    z-index: 1;  
}
```

Это будет GET-запрос.

Данные пользователей храните в json-файле.

Итого, ваш сервер должен обработать 1 POST, 2 PUT и 1 GET запросы.

`auth.html` будет стратовой, а `index.html` возвращает сервер.