<https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html>

Ajax là gì? Sử dụng Ajax jquery cho web app như thế nào?

Bài này thuộc phần 6 của 10 phần trong series [Tự học Javascript cơ bản](https://vntalking.com/series/tu-hoc-javascript-co-ban)

Nếu bạn đang [học Javascript](https://vntalking.com/series-tu-hoc-javascript-trong-10-tieng.html) thì thật sự đáng tiếc nếu bỏ qua khái niệm Ajax Jquery. Trước đây, mình chỉ biết Ajax là kỹ thuật làm cho web không thể SEO được. Thực sự thì mình rất ghét nó.

Nhưng thực ra, câu chuyện về Ajax nó còn nhiều thứ hay ho hơn rất nhiều. Nếu không có Ajax chắc sẽ không có khái niệm SPA (Single Page Aplication) như Gmail, Facebook, Twitter…

Vậy **Ajax là gì**? Tại sao công nghệ Ajax lại quan trọng đến vậy?

Bài viết này, chúng ta sẽ cùng nhau tìm hiểu Ajax trong Javascript và đặc biệt là **Ajax jquery**, thư viện đã giúp Ajax trở nên gần gũi hơn bao giờ hết.

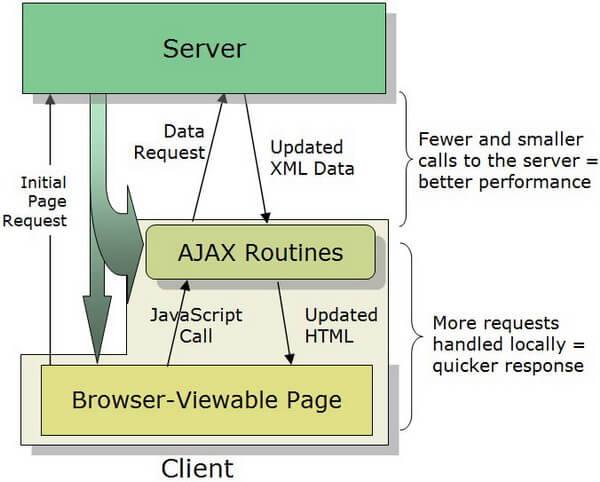
**Nội dung chính của bài viết**

* [#Ajax là gì?](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#ajax-la-gi)
* [#Jquery là gì? Ajax jquery là gì?](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#jquery-la-gi-ajax-jquery-la-gi)
* [#Một số ví dụ ứng dụng nổi tiếng sử dụng Ajax](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#mot-so-vi-du-ung-dung-noi-tieng-su-dung-ajax)
* [#Cách thức hoạt động của Ajax](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#cach-thuc-hoat-dong-cua-ajax)
  + [AJAX: Đồng bộ hay bất đồng bộ](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#ajax-dong-bo-hay-bat-dong-bo)
* [#Thực hành tạo request bằng Ajax](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#thuc-hanh-tao-request-bang-ajax)
  + [1. Viết Ajax với pure Javascript](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#1-viet-ajax-voi-pure-javascript)
  + [2. Viết Ajax Jquery (Ajax JS)](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#2-viet-ajax-jquery-ajax-js)
  + [#Tổng kết](https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html#tong-ket)

**#Ajax là gì?**

Đúng như với cái tên của nó, Ajax (Asynchronous JavaScript and XML) là một kỹ thuật giúp cho ứng dụng web hoạt động bất đồng bộ.

Với kỹ thuật này, bạn có thể tạo ra web động, nội dung thay đổi mà không cần phải reload lại trang. Chính vì bạn không phải load lại trang mà bạn hoàn toàn có thể thay đổi nội dung của từng phần trên web, giúp tiết kiệm tài nguyên mạng, web hoạt động mượt mà hơn.



Cơ chế hoạt động của Ajax jquery

Để hiểu cặn kẽ Ajax là gì, chúng ta cùng phân tích thuật ngữ trong tên gọi của nó:

* **Asynchronous**: Dịch lại là bất đồng bộ. Kỹ thuật bất đồng bộ tức là chương trình sẽ xử lý các hàm không theo một trình tự từ trên xuống. Chương trình cho tất cả các hàm được gọi vào một stack. Hàm nào xong trước thì trả kết quả trước, không ai phải chờ ai cả. Nếu bạn học Javascript, [Nodejs](https://vntalking.com/sach-hoc-lap-trinh-node-js-that-don-gian-html)… thì sẽ không lạ lẫm gì khái niệm này.
* **JavaScript**: Là ngôn ngữ lập trình để viết ra kỹ thuật Ajax.
* **XML**: Là định dạng dữ liệu theo kiểu tag, giống như HTML. Hiện nay có 2 kiểu dịnh dạng dữ liệu là JSON và XML. Cá nhân mình thì thích sử dụng [JSON](https://vntalking.com/json-la-gi-cach-parse-json-javascript-chuan-nhat.html) hơn vì nó dễ đọc hơn.

Hiện nay, hầu hết thư viện/framework Javascript đều hỗ trợ kỹ thuật Ajax. Nổi tiếng nhất là Jquery. Thế nên, nhiều bạn bị nhầm Ajax là độc quyền của Jquery, chỉ biết sử dụng Ajax bằng Jquery, tìm kiếm cũng chỉ biết jquery ajax…

Thui thì để cho đỡ lăn tăn, chúng ta cùng xem Jquery là gì mà nó lại nổi tiếng đến vậy.

**#Jquery là gì? Ajax jquery là gì?**

[Jquery](https://jquery.com/) là một thư viện Javascript mã nguồn mở, được thiết kế để phát triển các ứng dụng client (front-end).

Jquery được phát hành 2006 bởi lập trình viên lão luyện [John Resig](https://johnresig.com/), với triết lý: **Viết ít hơn – làm nhiều hơn**.

Thư viện này giúp đơn giản hóa tất cả các tác vụ của Javascript với HTML, xử lý sự kiện, tương tác với server qua Ajax…

Có lẽ Jquery đã trở thành thư viện không thể thiếu của bất kỳ website, ứng dụng web…Và rất nhiều framework sử dụng jquery như một phần core quan trọng như: [Bootstrap](https://getbootstrap.com/), [Vue](https://vuejs.org/), Angular…

Để bạn dễ hình dung, dưới đã là đoạn code cùng làm một việc nhưng được viết bằng Jquery và thuần Javascript:

//1. Animation fade in by Jquery

$(el).fadeIn();

//2. Animation fade in by pure Javascript

function fadeIn(el) {

  el.style.opacity = 0;

  var last = +new Date();

  var tick = function() {

    el.style.opacity = +el.style.opacity + (new Date() - last) / 400;

    last = +new Date();

    if (+el.style.opacity < 1) {

      (window.requestAnimationFrame && requestAnimationFrame(tick)) || setTimeout(tick, 16);

    }

  };

  tick();

}

fadeIn(el);

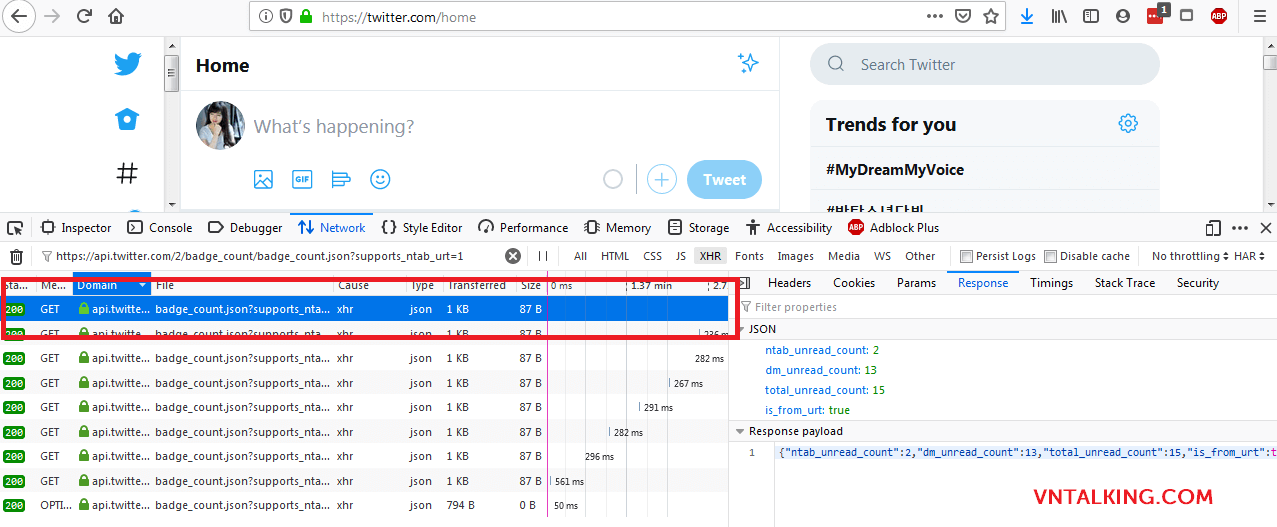
Bạn thấy sự khác biệt chưa?

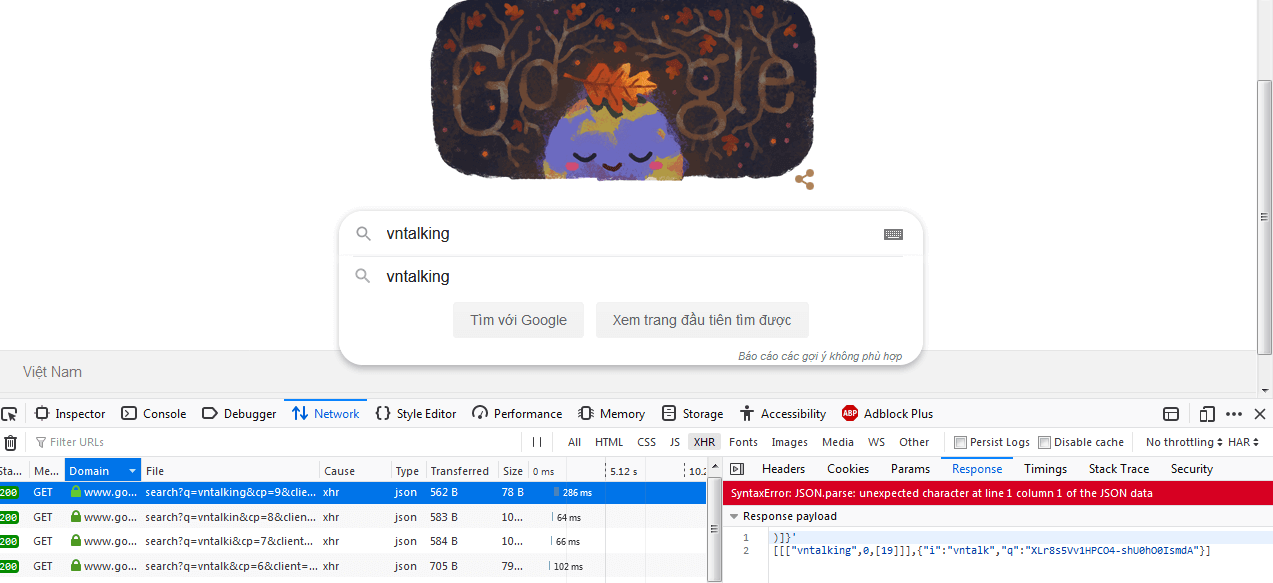
**Một phút cho quảng cáo:**Nói đến xử lý bất đồng bộ, mà lại không hiểu về Promise thì thật là thiết sót. Mời bạn đọc bài viết: [Promise là gì? Xử lý bất đồng bộ trong JS](https://vntalking.com/cach-su-dung-promise-trong-javascript.html)

**#Một số ví dụ ứng dụng nổi tiếng sử dụng Ajax**

Để các bạn có thể thấy được sự nổi tiếng và tầm quan trọng của kỹ thuật Ajax, mình sẽ lấy một số ứng dụng nổi tiếng.

**1. Lấy số lượng các thông báo trênTwitter:**  
Theo như mình thấy, có vẻ như Twitter sử dụng Ajax để lấy số lượng thông báo mà người dùng chưa đọc. Bạn xem ảnh bên dưới:

**2. Tính năng gợi ý từ khóa của Google Search**  
Chắc hẳn bạn sử dụng tính năng này hàng ngày ấy nhỉ! Nhưng bạn có để ý là Google cũng sử dụng Ajax để tạo request tới server mà không cần phải tải lại trang không?

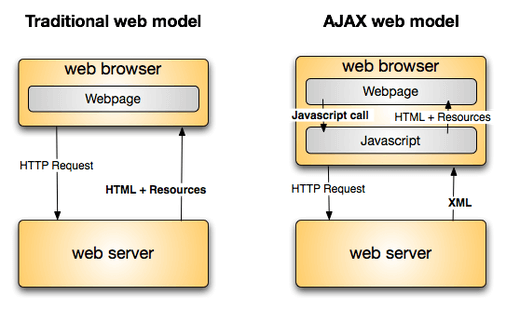


Ý tưởng của AJAX đã manh nha từ những năm 90 của thế kỷ trước rồi. Chỉ đến khi Google, ứng dụng kỹ thuật Ajax vào các sản phẩm nổi tiếng như Google Map, Google Mail, Google Maps… thì nó mới được công nhận rộng rãi.

Ngày nay, Ajax đã quá phổ biến rồi, với rất nhiều ứng dụng web sử dụng nó để đơn giản quá trình trao đổi với server.

**#Cách thức hoạt động của Ajax**

Như mình đã nói ở trên, Ajax không phải là một ngôn ngữ lập trình, ajax là một bộ kỹ thuật kết hợp nhiều công nghệ và được thực hiện bởi Javascript.



Cách thức hoạt động của Ajax

Nhìn vào sơ đồ trên, bạn có nhận ra sự khác biệt cơ bản giữa một request thông thường với một request Ajax không?

Đó là: Thay vì server trả về cả một trang web (gồm HTML + resource CSS, Image…) thì server chỉ cần trả về dữ liệu cần thiết mà thôi.

**AJAX: Đồng bộ hay bất đồng bộ**

Mặc dù trong tên gọi của nó có chữ bất đồng bộ (asynchronously). Nhưng thực tế kỹ thuật Ajax lại hỗ trợ cả đồng bộ và bất đồng bộ.

Trong một số trường hợp, bạn cần phải tạo request đồng bộ, nhưng yêu cầu người dùng phải đợi request hoàn thành mới được làm việc tiếp theo.

Để bật tắt chế độ đồng bộ hay bất đồng bộ trong **Ajax Jquery** rất đơn giản, chỉ cần thay đổi giá trị boolean cờ async. Mình sẽ giới thiệu cách thực hiện ở phần tiếp của bài viết nhé.

**#Thực hành tạo request bằng Ajax**

Sau khi đã hiểu Ajax là gì, phần này chúng ta sẽ thực hành viết code nhé.

Để cho các bạn có cái nhìn đa chiều, mình sẽ hướng dẫn các bạn tạo một request Ajax bằng cả pure javascript và jquery. Sau đó, bạn thử xem cách nào hay hơn nhé.

Bài viết ví dụ ở đây là: mình sẽ tạo một request GET tới một REST API, và server sẽ trả lại data dưới dạng JSON.

**Có thể bạn quan tâm:**Ví dụ này mình có tạo request tới một REST API, vậy bạn đã biết cách REST API trên server bằng Javascript chưa? Của bạn đây nhé: [Cách tạo REST API đơn giản với NodeJS](https://vntalking.com/tao-restful-api-bang-nodejs-mongodb.html)

**1. Viết Ajax với pure Javascript**

function ajax\_get(url, callback) {

    var xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {

        if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {

            callback(xmlhttp.responseText);

        }

    };

    xmlhttp.open("GET", url, true);

    xmlhttp.send();

}

ajax\_get('https://reqres.in/api/users', function(data) {

   var json = JSON.parse(data);

      console.log(json);

});

♥ Đọc ngay: [**Parse Json Javascript**](https://vntalking.com/json-la-gi-cach-parse-json-javascript-chuan-nhat.html)

**2. Viết Ajax Jquery (Ajax JS)**

function get(url, callback) {

    $.ajax({

      url: url,

      type: "GET",

      async: true,

      success: function(response) {

        callback(response);

      }

    });

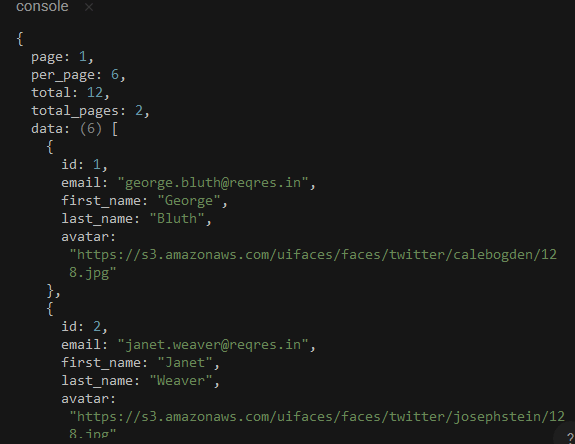
  }

  get('https://reqres.in/api/users', function(data) {

    console.log(data);

  });

Cả hai cách đều trả về kết quả là dữ liệu dạng JSON. Và tất nhiên Jquery đã parse JSON hộ mình luôn.



Nhìn chung thì cách viết Ajax Jquery vẫn đơn giản và nhìn dễ hiểu hơn nhiều đúng không?

<https://playcode.io/457423>

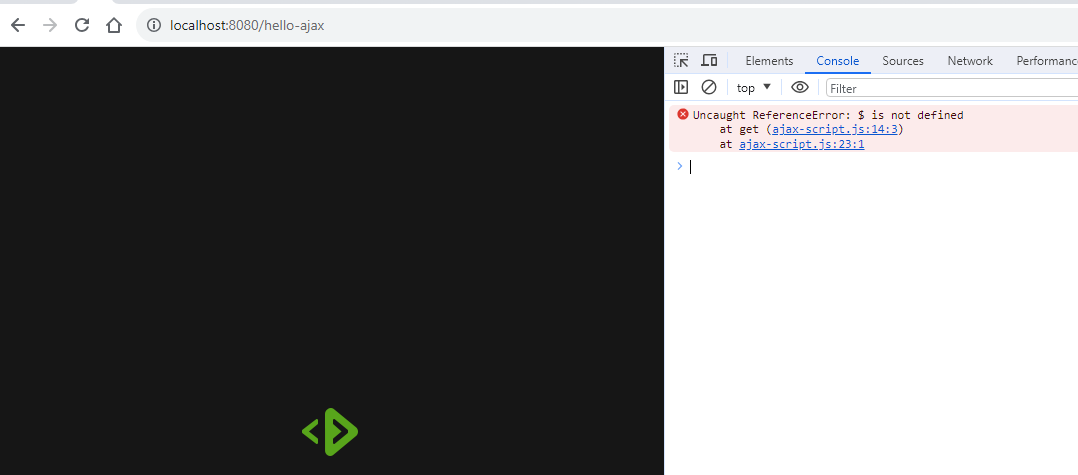
### BEGIN INSTRUCTION ###

Fix error in ajax-script.js when browser render ajax.html

Uncaught ReferenceError: $ is not defined

at get (ajax-script.js:14:3)

at ajax-script.js:23:1



### BEGIN CONTEXT ###'

# BEGIN ajax.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Test AJAX</title>

</head>

<body>

<div>

<img src="https://playcode.io/static/img/logo.png"

alt="PlayCode logo">

<h1 id="msg"></h1>

</div><!--Please note that the project you are working on is a legacy project which may not function correctly. In order to improve its performance, there are two key actions that need to be taken:

1. The first action to take is to import packages in JavaScript files as the global libs are no longer available.

2. The second action is to verify the file imports, as they may be broken. It is important to ensure that all files are properly imported to avoid any potential issues.-->

<link rel="stylesheet" href="css/ajax-styles.css" />

<script src="scripts/ajax-script.js"></script>

</body>

</html>

# END ajax.html

### END CONTEXT ###

### BEGIN INPUT ###

# BEGIN scripts/ajax-script.js

function ajax\_get(url, callback) {

    var xmlhttp = new XMLHttpRequest();

    xmlhttp.onreadystatechange = function() {

      if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {

        callback(xmlhttp.responseText);

      }

    };

    xmlhttp.open("GET", url, true);

    xmlhttp.send();

  }

  function get(url, callback) {

    $.ajax({

      url: url,

      type: "GET",

      success: function(response) {

        callback(response);

      }

    });

  }

  get('https://reqres.in/api/users', function(data) {

    console.log(data);

  });

# END scripts/ajax-script.js

### END INPUT ###

Ans:

The error you're encountering, "Uncaught ReferenceError: $ is not defined," typically occurs when jQuery is being used but hasn't been properly included in the project. In your **ajax-script.js**, the function **get** uses jQuery's **$.ajax** method, but it appears that jQuery is not imported in your **ajax.html**.

To fix this issue, you will need to include jQuery in your HTML file before your script file. You can do this by adding a **<script>** tag that points to a CDN hosting the jQuery library. Here's how you can modify your **ajax.html**:

**Modified ajax.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Test AJAX</title>

    <link rel="stylesheet" href="css/ajax-styles.css" />

</head>

<body>

<div>

    <img src="https://playcode.io/static/img/logo.png"

         alt="PlayCode logo">

    <h1 id="msg"></h1>

</div>

<!-- Include jQuery from a CDN -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script src="scripts/ajax-script.js"></script>

</body>

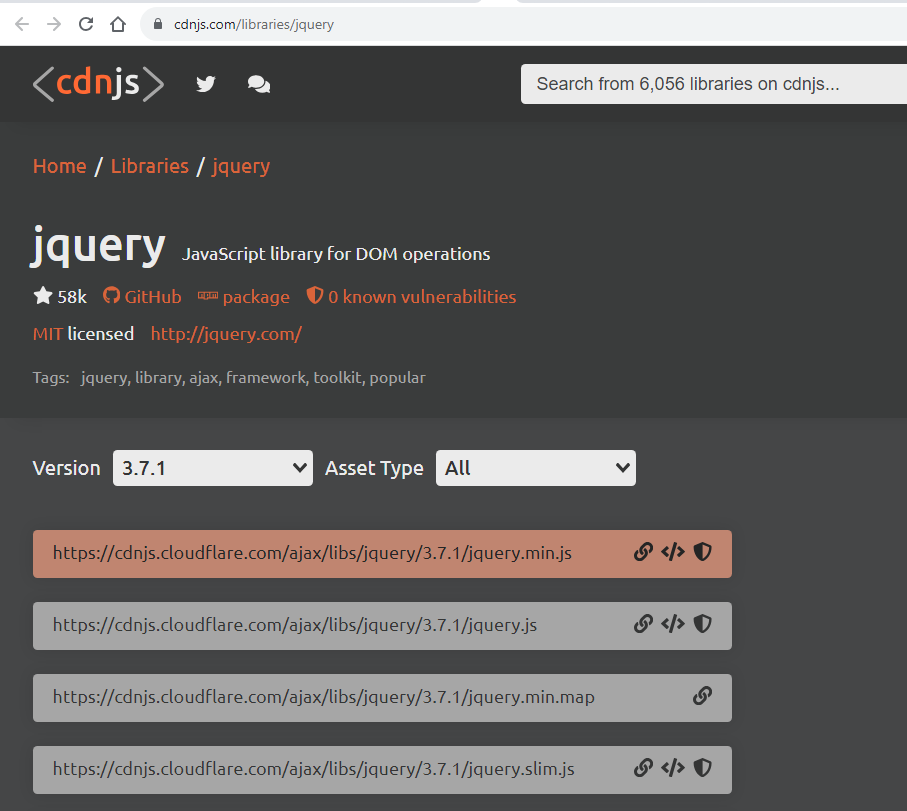
</html>

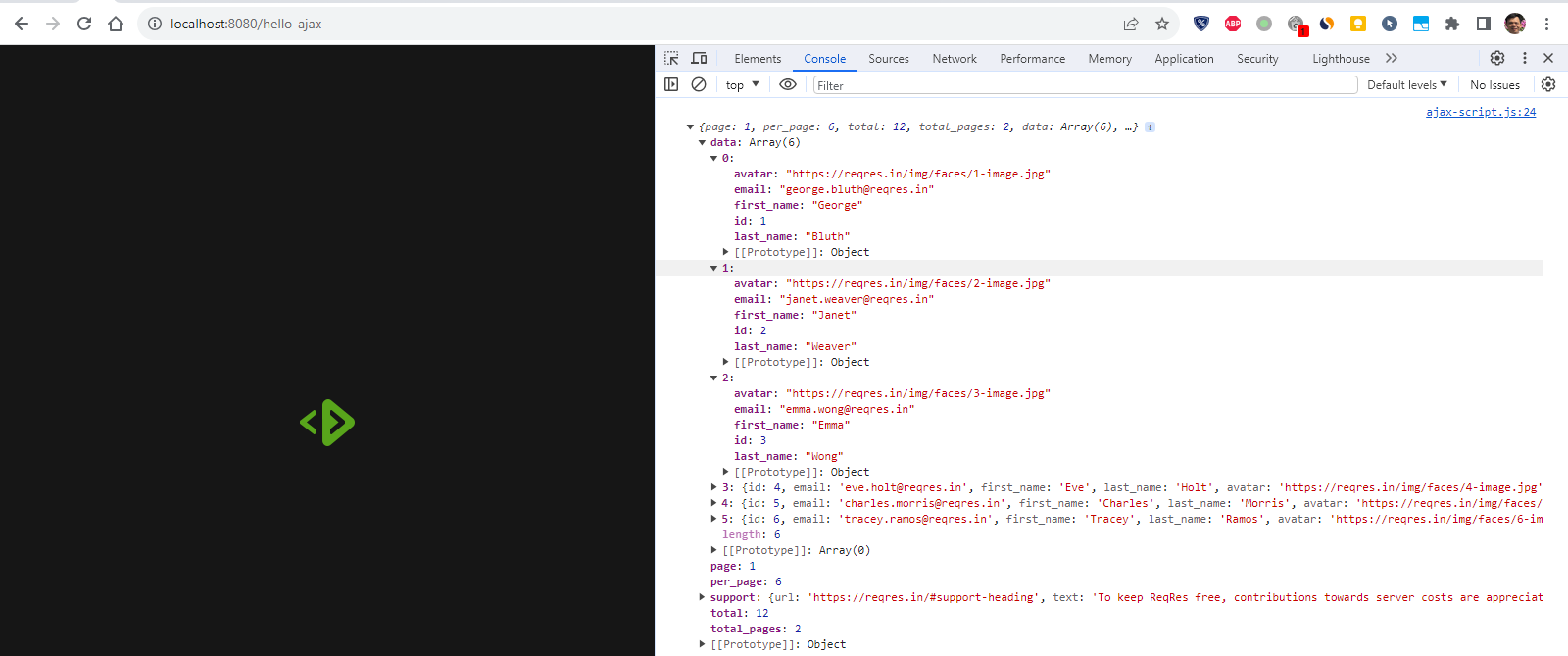
**Explanation:**

1. **Include jQuery**: The line **<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>** includes jQuery in your HTML file. This should be placed before the inclusion of your custom script **ajax-script.js**.
2. **Script Order**: Make sure that the jQuery script tag is before your custom script tag. This ensures that jQuery is loaded and available for use in your custom script.

After making this change, the jQuery functions used in your **ajax-script.js** file should work correctly without the "Uncaught ReferenceError" issue. Remember to always include dependencies like jQuery before they are used in your custom scripts.

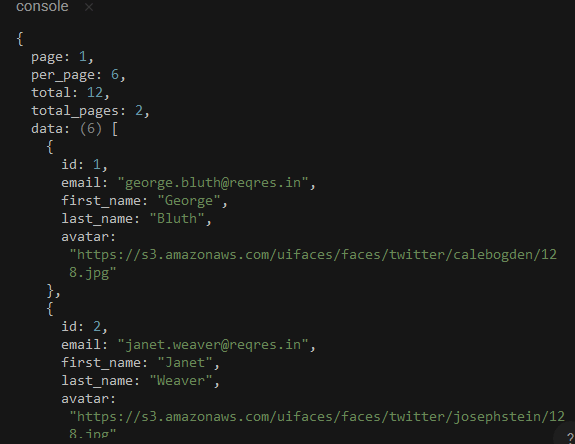
<https://cdnjs.com/libraries/jquery>





<https://vntalking.com/ajax-la-gi-su-dung-ajax-jquery-cho-web-app-nhu-the-nao.html>

Cả hai cách đều trả về kết quả là dữ liệu dạng JSON. Và tất nhiên Jquery đã parse JSON hộ mình luôn.



Nhìn chung thì cách viết Ajax Jquery vẫn đơn giản và nhìn dễ hiểu hơn nhiều đúng không?

To modify your **ajax.html** file so that it can display the returned JSON data in a table, you'll need to add an empty table structure in your HTML and then modify your JavaScript code to populate this table with the data received from the AJAX request. Here's how you can do it:

**Modified ajax.html:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Hello AJAX JQuery</title>

    <link rel="stylesheet" href="css/ajax-styles.css" />

</head>

<body>

<div>

    <img src="https://playcode.io/static/img/logo.png" alt="PlayCode logo">

    <h1 id="msg"></h1>

    <table id="userTable">

        <thead>

            <tr>

                <th>ID</th>

                <th>Email</th>

                <th>First Name</th>

                <th>Last Name</th>

                <th>Avatar</th>

            </tr>

        </thead>

        <tbody>

            <!-- Data rows will be added here dynamically -->

        </tbody>

    </table>

</div>

<!-- Include jQuery -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

<script src="scripts/ajax-script.js"></script>

</body>

</html>

**Modified ajax-script.js:**

function ajax\_get(url, callback) {

    // ... existing code ...

  }

  function get(url, callback) {

    // ... existing code ...

  }

  get('https://reqres.in/api/users', function(data) {

    var users = JSON.parse(data).data; // Assuming the response is JSON string

    var tableBody = $('#userTable tbody');

    users.forEach(function(user) {

      var row = '<tr>' +

                  '<td>' + user.id + '</td>' +

                  '<td>' + user.email + '</td>' +

                  '<td>' + user.first\_name + '</td>' +

                  '<td>' + user.last\_name + '</td>' +

                  '<td><img src="' + user.avatar + '" alt="' + user.first\_name + '"></td>' +

                '</tr>';

      tableBody.append(row);

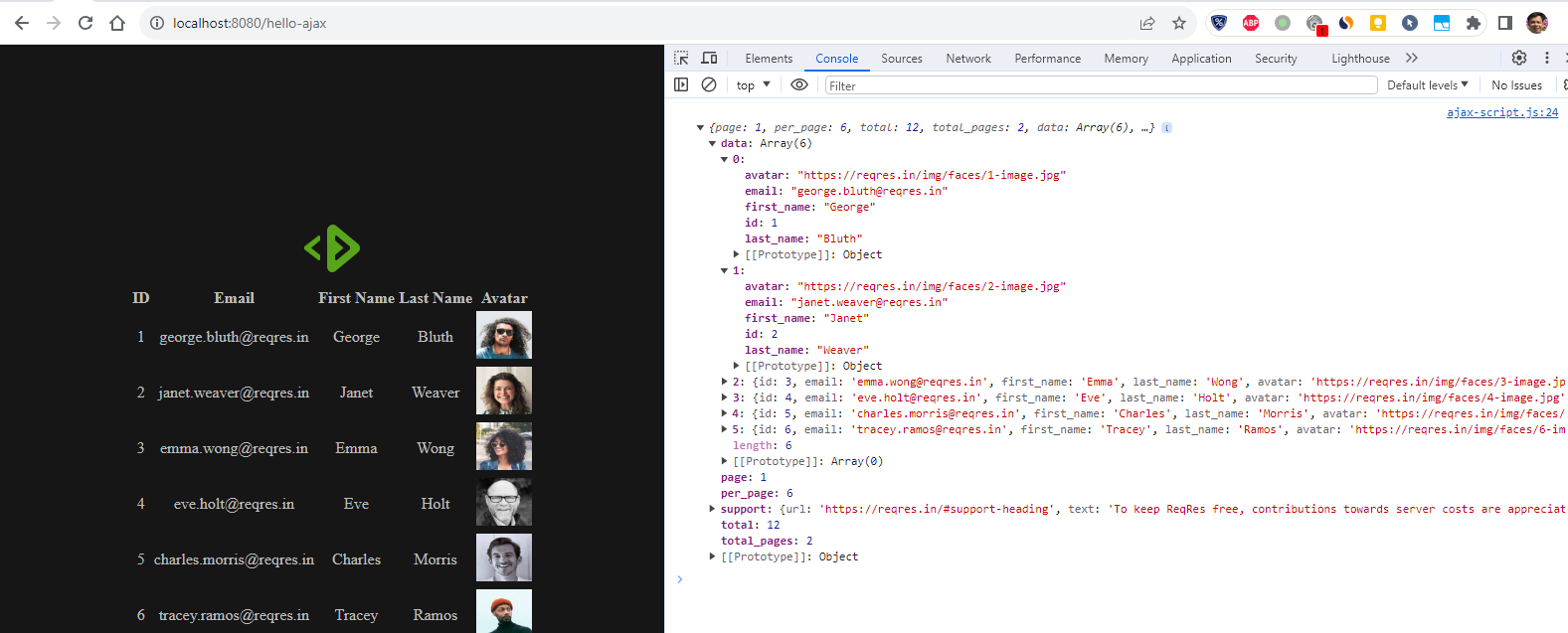
    });

  });

**Explanation:**

1. **HTML Table**: The **ajax.html** file now includes a basic HTML table structure with **thead** for headers and an empty **tbody** where the data rows will be dynamically added.
2. **Populating the Table**: The **ajax-script.js** script is modified to parse the returned JSON data and loop through each user object. For each user, a new row (**<tr>**) is constructed with their data and appended to the table body.
3. **Dynamic Data Insertion**: The script dynamically inserts the user data into the table, creating a row for each user in the returned JSON data.

With these modifications, when your page makes the AJAX request to the specified URL, the returned data will be displayed in a table format on your **ajax.html** page. Ensure your AJAX request is correctly fetching data and that the structure of the received JSON matches the parsing logic in your JavaScript.



### BEGIN INSTRUCTION ###

Fix the below code line in ajax-script.js. Reason JQuery library has already parse JSON for data variable

var users = JSON.parse(data).data; // Assuming the response is JSON string

# BEGIN ERROR MESSAGE ON CONSOLE

VM202:1 Uncaught SyntaxError: "[object Object]" is not valid JSON

at JSON.parse (<anonymous>)

at ajax-script.js:25:20

at Object.success (ajax-script.js:18:7)

at c (jquery.min.js:2:25304)

at Object.fireWith [as resolveWith] (jquery.min.js:2:26053)

at l (jquery.min.js:2:77782)

at XMLHttpRequest.<anonymous> (jquery.min.js:2:80265)

# END ERROR MESSAGE ON CONSOLE

The error you're encountering in your **ajax-script.js** is because jQuery's **$.ajax** method automatically parses the JSON response if the **Content-Type** header of the response is set to **application/json**. Therefore, attempting to parse the already parsed **data** object using **JSON.parse** results in an error.

To fix this, you simply need to remove the **JSON.parse** call and use the **data** object directly, as it's already a JavaScript object. Here's the corrected line in your script:

**Corrected Line in ajax-script.js:**

javascriptCopy code

// Remove JSON.parse as the data is already a JavaScript object

get('https://reqres.in/api/users', function(data) {

    console.log(data);

    var users = data.data; // Directly use the data object

    // ... rest of the code ...

  });

This change should resolve the **SyntaxError** you're encountering, as **data** is now correctly treated as an object rather than a JSON string.

### BEGIN INSTRUCTION ###

Check if is there any incorrect in the ajax-script.js. Currently, after I click the button "Create user", I found that there are 2 HTTP requests instead of 1 HTTP request (1st HTTP request)

1st HTTP request

curl "https://reqres.in/api/users" ^

-H "authority: reqres.in" ^

-H "accept: \*/\*" ^

-H "accept-language: en,vi;q=0.9,uz;q=0.8" ^

-H "content-type: application/json" ^

-H "origin: http://localhost:8080" ^

-H "referer: http://localhost:8080/" ^

-H "sec-ch-ua: ^\^"Google Chrome^\^";v=^\^"119^\^", ^\^"Chromium^\^";v=^\^"119^\^", ^\^"Not?A\_Brand^\^";v=^\^"24^\^"" ^

-H "sec-ch-ua-mobile: ?0" ^

-H "sec-ch-ua-platform: ^\^"Windows^\^"" ^

-H "sec-fetch-dest: empty" ^

-H "sec-fetch-mode: cors" ^

-H "sec-fetch-site: cross-site" ^

-H "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36" ^

--data-raw "^{^\^"name^\^":^\^"morpheus^\^",^\^"job^\^":^\^"leader^\^"^}" ^

--compressed

2nd HTTP request

curl "https://reqres.in/api/users" ^

-X "OPTIONS" ^

-H "authority: reqres.in" ^

-H "accept: \*/\*" ^

-H "accept-language: en,vi;q=0.9,uz;q=0.8" ^

-H "access-control-request-headers: content-type" ^

-H "access-control-request-method: POST" ^

-H "origin: http://localhost:8080" ^

-H "referer: http://localhost:8080/" ^

-H "sec-fetch-dest: empty" ^

-H "sec-fetch-mode: cors" ^

-H "sec-fetch-site: cross-site" ^

-H "user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/119.0.0.0 Safari/537.36" ^

--compressed

### BEGIN CONTEXT ###

### END CONTEXT ###

### BEGIN INPUT ###

# BEGIN WEB-INF/ajax.html

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<title>Hello AJAX JQuery</title>

<link rel="stylesheet" href="css/ajax-styles.css" />

</head>

<body>

<div>

<img src="https://playcode.io/static/img/logo.png" alt="PlayCode logo">

<h1 id="msg"></h1>

<button id="createUserBtn">Create User</button> <!-- Add this line -->

<table id="userTable">

<thead>

<tr>

<th>ID</th>

<th>Email</th>

<th>First Name</th>

<th>Last Name</th>

<th>Avatar</th>

</tr>

</thead>

<tbody>

<!-- Data rows will be added here dynamically -->

</tbody>

</table>

</div>

<!-- Include jQuery -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

<script src="scripts/ajax-script.js"></script>

</body>

</html>

# END WEB-INF/ajax.html

# BEGIN scripts/ajax-script.js

function ajax\_get(url, callback) {

var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function() {

if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {

callback(xmlhttp.responseText);

}

};

xmlhttp.open("GET", url, true);

xmlhttp.send();

}

function createUser() {

$.ajax({

url: 'https://reqres.in/api/users',

type: 'POST',

contentType: 'application/json',

data: JSON.stringify({ "name": "morpheus", "job": "leader" }),

success: function(response) {

console.log('User Created:', response);

// Handle the response here (e.g., update the UI)

},

error: function(xhr, status, error) {

console.error('Error in user creation:', status, error);

}

});

}

$(document).ready(function() {

$('#createUserBtn').click(createUser);

});

function get(url, callback) {

$.ajax({

url: url,

type: "GET",

success: function(response) {

callback(response);

}

});

}

get('https://reqres.in/api/users', function(data) {

console.log(data);

var users = data.data;

var tableBody = $('#userTable tbody');

users.forEach(function(user) {

var row = '<tr>' +

'<td>' + user.id + '</td>' +

'<td>' + user.email + '</td>' +

'<td>' + user.first\_name + '</td>' +

'<td>' + user.last\_name + '</td>' +

'<td><img src="' + user.avatar + '" alt="' + user.first\_name + '"></td>' +

'</tr>';

tableBody.append(row);

});

});

# END scripts/ajax-script.js

### END INPUT ###

Ans1:

The JavaScript code in **ajax-script.js** appears to be correctly formatted for making an AJAX POST request to create a user. The issue of two HTTP requests being made — one OPTIONS and one POST — is related to how browsers handle cross-origin requests (CORS) rather than an error in your JavaScript code.

**Understanding the Behavior:**

1. **OPTIONS Request**: This is a preflight request sent by the browser. It's a CORS mechanism to ensure that the server accepts the actual request from the origin your script is running on. The browser does this automatically before sending the POST request.
2. **POST Request**: This is the actual request you intended to send, as defined in the **createUser()** function.

**Code Review:**

Your **createUser** function looks correct for sending a POST request:

* It specifies the URL, request type (POST), content type (application/json), and the data payload.
* It includes success and error handlers to process the response or any errors.

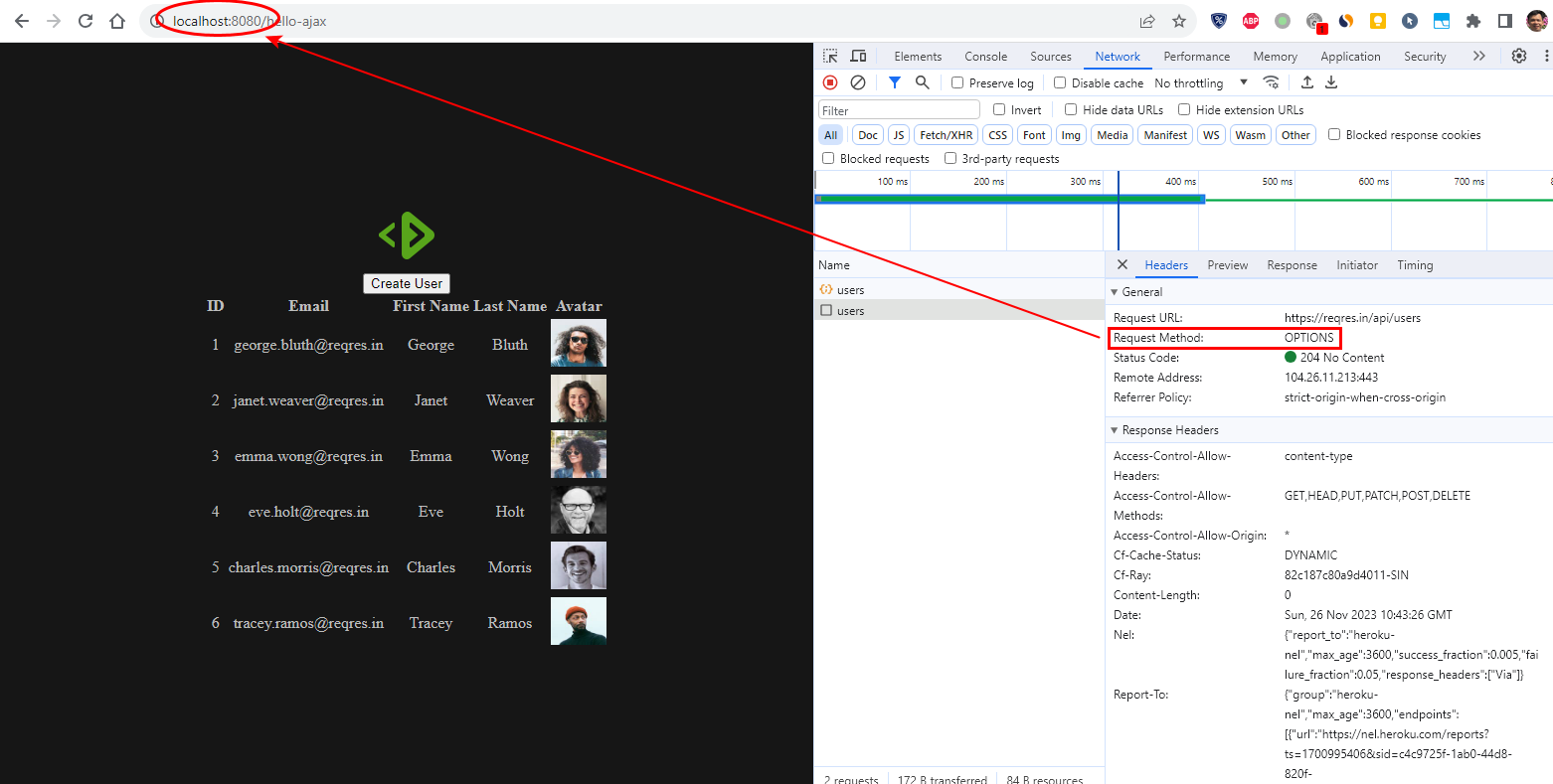
**Possible Adjustments:**

If you're concerned about the OPTIONS request and it's not just a matter of understanding the browser behavior, you could consider the following:

* **Server-Side Handling**: Ensure the server is configured to handle CORS correctly. It should respond to the OPTIONS request with appropriate CORS headers, allowing the POST request from your origin.
* **Same-Origin Policy**: If possible, host your JavaScript and API on the same domain to avoid triggering CORS.
* **Proxy Server**: As a workaround, use a server-side proxy on the same domain to relay requests to the external API.

**Conclusion:**

The JavaScript code you've written is standard for AJAX requests, and the appearance of an OPTIONS request is a normal browser behavior in the context of CORS. Unless there are specific issues with how the server handles these requests, there's no need to modify your JavaScript code for this concern.



Nguyên nhân: là ta request 'https://reqres.in/api/users' khác với domain localhost:8080

