JavaScript: calculate the difference between two dates in human format

Implementing a JavaScript function to calculate the time elapsed from a date in human format can be very useful in several applications. This feature can be used to show how much time has passed since a specific event, such as a social media post or a product update in your online store. In this article, I'll show you how to create a simple and effective function to calculate elapsed time legibly.

First of all, let's define the goal of our function. We want it to input a date in JavaScript format and return the elapsed time since the current date in a human understandable format. For example, if the input date is "2023-06-15", the function should return a string like "2 weeks ago" or "1 month ago".

Let's start by writing our function:

```
function getElapsedTime(date) {
   const currentDate = new Date();
   const initialDate = new Date(date);

   const timeDifferenceInMilliseconds = currentDate -
initialDate;
   const timeDifferenceInSeconds =
Math.floor(timeDifferenceInMilliseconds / 1000);
   const timeDifferenceInMinutes =
Math.floor(timeDifferenceInSeconds / 60);
   const timeDifferenceInHours =
Math.floor(timeDifferenceInMinutes / 60);
   const timeDifferenceInDays =
```

```
Math.floor(timeDifferenceInHours / 24);
   const timeDifferenceInWeeks =
Math.floor(timeDifferenceInDays / 7);
   const timeDifferenceInMonths =
Math.floor(timeDifferenceInDays / 30);
   const timeDifferenceInYears =
Math.floor(timeDifferenceInDays / 365);

   if (timeDifferenceInYears > 0) {
     return `${timeDifferenceInYears}
year${timeDifferenceInYears > 1 ? 's' : ''} ago`;
   } else if (timeDifferenceInMonths > 0) {
     return `${timeDifferenceInMonths}
month${timeDifferenceInMonths > 1 ? 's' : ''} ago`;
   } else if (timeDifferenceInWeeks > 0) {
     return `${timeDifferenceInWeeks}
week${timeDifferenceInWeeks > 1 ? 's' : ''} ago`;
   } else if (timeDifferenceInDays > 0) {
     return `${timeDifferenceInDays}
day${timeDifferenceInDays > 1 ? 's' : ''} ago`;
   } else if (timeDifferenceInHours > 0) {
     return `${timeDifferenceInHours}
hour${timeDifferenceInHours > 1 ? 's' : ''} ago`;
   } else if (timeDifferenceInMinutes > 0) {
     return `${timeDifferenceInMinutes}
minute${timeDifferenceInMinutes > 1 ? 's' : ''} ago`;
   } else {
     return `${timeDifferenceInSeconds}
second${timeDifferenceInSeconds > 1 ? 's' : ''} ago`;
   }
}
```

In our function, we get the current date using new Date(). Next, we create a Date object from the starting date given as input. We then calculate the

difference between the two dates in milliseconds using the subtraction operator -.

Next, we convert the difference in milliseconds to seconds, minutes, hours, days, weeks, months, and years. We use the Math.floor function to round the result to the nearest lower integer. This allows us to get the elapsed time in whole numbers instead of decimals.

Finally, we use a series of if-else statements to determine the correct format to return based on the amount of time that has elapsed. If more than 365 days have passed, we return the past year. If more than 30 days have passed, we return the number of months, and so on. If none of the previous conditions are satisfied, we return the number of seconds that have elapsed.

Here is an example of how to use our function:

```
const initialDate = '2023-06-15';
const elapsed_time = getElapsedTime(initialDate);
console.log(elapsed_time); // Output: "2 weeks ago"
```

With this simple function, you can calculate and legibly display elapsed time since a specific date. You can further tailor the feature to fit your specific needs, for example by changing the output formats or adding custom conditions.