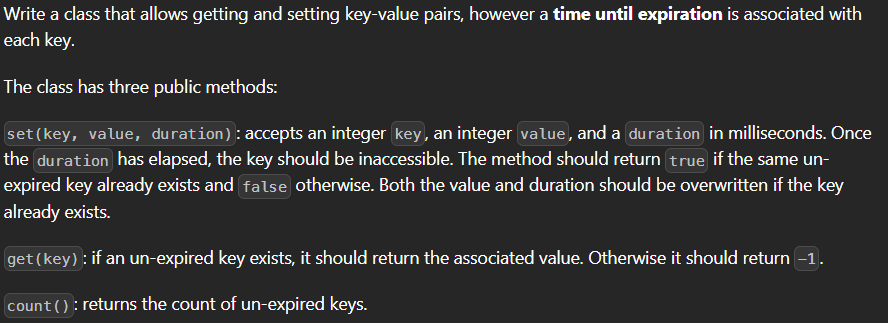
Problem



Solution

var TimeLimitedCache = function() {

    this.cache = new Map();

};

/\*\*

 \* @param {number} key

 \* @param {number} value

 \* @param {number} duration time until expiration in ms

 \* @return {boolean} if un-expired key already existed

 \*/

TimeLimitedCache.prototype.set = function(key, value, duration) {

    const now = Date.now();

    if (this.cache.has(key)) {

        this.cache.set(key, [value, now + duration]);

        return true;

    }

    this.cache.set(key, [value, now + duration]);

    return false;

};

/\*\*

 \* @param {number} key

 \* @return {number} value associated with key

 \*/

TimeLimitedCache.prototype.get = function(key) {

    const now = Date.now();

    if (this.cache.has(key)) {

        const [value, expiry] = this.cache.get(key);

        if (now < expiry) {

            return value;

        } else {

            this.cache.delete(key);

        }

    }

    return -1;

};

/\*\*

 \* @return {number} count of non-expired keys

 \*/

TimeLimitedCache.prototype.count = function() {

    const now = Date.now();

    let count = 0;

    for (const [key, [\_, expiry]] of this.cache) {

        if (now < expiry) {

            count++;

        } else {

            this.cache.delete(key);

        }

    }

    return count;

};