JavaScript Coding Puzzle (BE2)

Slow API Challenge

Introduction

AllTheRooms accesses several slow APIs, and effective caching is critical.

Assume the following functions exist:

```
// stores data (value) by key
async function cache_store(key, value) {
}

// retrieves data by key (if it exists)
async function cache_retrieve(key) {
}

// fetches data from a slow data source
async function slow_function(input) {
}
```

For the challenge

Your job is to speed up `slow_function` by completing the `memoize` function below. Speed is absolutely critical and therefore the results are expected to be returned as soon as information is ready.

```
// runs faster than slow_function by using cache functions
function memoize(slow_function) {
      return fast_function;
}
```

The input of `memoize` is `slow_function` and the output is a faster function.

Sometimes the cache function can be slower than the `slow_function`. Your `memoize` implementation must:

1. Cache the result of `slow_function` using the caching functions.
2. Return the fastest: the cached result or the fresh result. This means if the cache retrieval completes first, then that result should be returned, and if the `slow_function` completes first, then that result should be returned. This will result in the fastest possible version of `fast_function`.
3. Since we are always calling `slow_function`, we should always update the cache in either scenario.

Bonus question:

If cached values have an accuracy half-life of 1000 seconds, what is the TTL to achieve 95% accuracy?

Please email any questions and also the link and code. Thank you!