## Goal

Create a small web service that can produce identifiers that are unique within a given namespace.

## Solution

Solution of this problem highly depends on the context where this service will be used. Without knowing the context I’ll provide two possible solutions and will describe the pros and cons of each of them

### Random IDs

Modern cryptography and hash functions are capable to generate values with probability of collision about . For example for SHA-256 (n=256) and one billion generations () probability is about . This approach does not guarantee that every ID will be unique but chances to generate two similar IDs are extremely low. In most cases I would go with this approach and best of all it does not require to build any service, it can be as small as a single function, with predictable behavior.

#### Pros of this approach

1. Does not require any dependencies. Most modern languages have cryptographic module.
2. Predictable behaviour
3. Scales with host service

#### Cons of this approach

1. Doesn’t guarantee uniqueness of ID, but the probability of collision is very low.
2. Can’t track all requested namespaces and total number of issued IDs

### Serial IDs

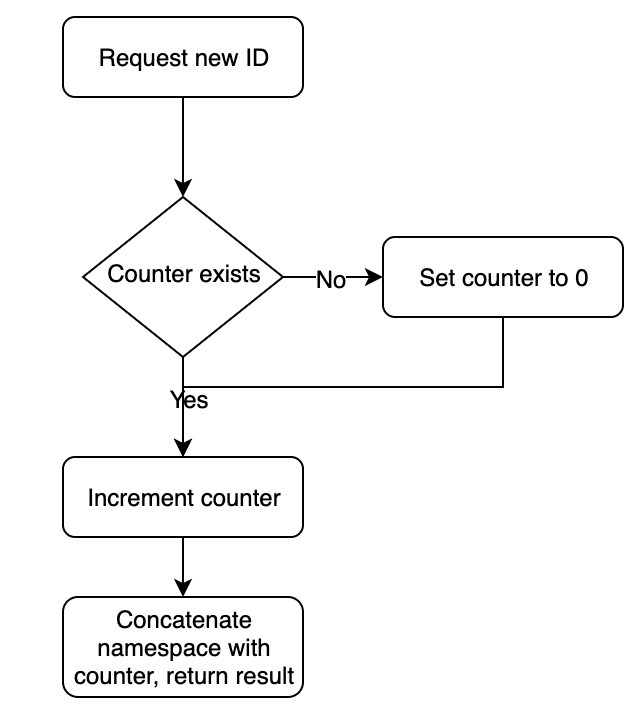
This one is a bit more complicated because it can’t be purely stateless and requires some form of dependencies that will store whole sequence or just last value.

In my case I’ve decided to use Redis with persistent storage. That’s in-memory database which is relatively fast. Can work as a cluster. Redis is single-threaded and guarantees that each command will be processed by the server one at a time in an atomic manner, so in our case it should be safe of race-condition.

As a web server I’ve decided to use express, that’s defacto standard web-server in JS world, there are better alternatives, with better performance, but this one should do fine.

Here I’ll describe the steps of how a new ID is generated, and below you can find flow chart as an illustration.

1. GET <http://localhost:3000/newId/foo>
2. Send INCR request to Redis to increment counter for **foo** namespace
3. Redis verifies if counter for key **foo** exists, if it doesn’t it sets it to 0, and then increment
4. Server waits Redis to return counter value, then concatenate it with namespace and return result.



#### Pros of this approach

1. Guaranteed to generate unique IDs
2. Can get list and status of each generated namespace

#### Cons of this approach

1. Has dependency of another service (Redis)
2. Scales, but requires more infrastructure.
3. Potentially has greater latency