Policy Enforcement Assessment: Stage 2

Definitions

Each policy has the following fields:

- `name` - a textual name, consisting of at most 32 alphanumeric characters and underscores
- `description` - free text
- `type` - either of the strings `"Arupa"` or `"Frisco"`

An Arupa policy may not have the same `name` as another Arupa policy. Multiple Frisco policies may have the same `name`. This requirement replaces the global policy name uniqueness requirement from the previous step.

Requirements

Building on the previous stage, implement read, update, and delete, methods for policies. All methods should continue to take and return JSON strings and raise exceptions.

- The `read_policy()`, `update_policy()`, and `delete_policy()` methods should take as their first argument the same object identifiers which the `create_policy()` method returns.
- The `read_policy()` method should return a JSON string containing an object in a format similar or identical to the objects in the response from the `list_policies()` method.
- The `update_policy()` method should take as its second argument the same input which is passed to the `create_policy()` method, supporting the same set of fields.

Other Instructions

The problem specification may be intentionally vague or incomplete.

- If you encounter something which was under-specified, make reasonable assumptions about the correct behavior.
- If you're unsure which assumption to make, pick the one which requires the least amount of work to implement.
- **It's important that you document all assumptions you make and communicate them clearly in your submission.**