iModelHub team

TypeScript and node.js coding task

Description

Implement a simple service responsible for routing requests and rate-limit them based on `clientId` value set in the header. Rate-limiting and routing will be configurable via requests to `/configure` endpoint.
There also should be tests which demonstrates usage of the API gateway service.
Non-functional requirements will also be taken into account e.g.: maintainability of the repository.

Route entity:
- `sourcePath` a path for a redirect e.g.: `/items`. `sourcePath` cannot start with `/configure`.
- `destinationUrl` an URL to redirect to e.g.: `https://example.com/items`

Client entity:
- `clientId` - sets the expected client-id mapping from the request header
- `limit` - how many calls can be made per seconds defined. If not provided it should default to 1.
- `seconds` - how many seconds per limit are defined. If not provided it should default to 1.
- `limit` and `seconds` are optional and of type number while `clientId` is a mandatory string.

Your service should have APIs to:
- bulk create routes and client configurations
- list all the configured routes and their rate-limits

Check for client-id in the request header:
- each request requires client-id header
- if client-id is not provided then the 400 status code must be returned
- if client-id is provided but not found in configuration then the 403 status code must be returned
- if a client has exceeded their limit then return the 429 status code

Check for requested path in routes `sourcePath` configuration:
- if matched `sourcePath` then set location in the header to destinationUrl from matching entry in the configuration and return the 302 status code
- if no matched `sourcePath` is found then the 404 status code must be returned

Rate-limiting
"In computer networks, rate limiting is used to control the rate of requests sent or received by a network interface controller and is used to prevent DoS attacks."
- requests to the service should be counted except for the /configure endpoint.
- /configure endpoint should be accessible only by the set of clients defined at deployment time
- store number of the requests made for each clientId for the purpose of rate-limiting
- if a client has exceeded their limit then return the 429 status code.

Example

```
POST /configuration HTTP/1.1
{
  "routes":[
    {
      "sourcePath": "/items",
      "destinationUrl": "https://example.com/items"
    }
  ],
  "clients":[
    {
      "clientId": "1234",
       "limit": 1000,
       "seconds": 10
    }
  ]
}
```

This payload will configure the API Gateway so that it will:

- configure `/items` requests to be routed to `https://example.com/items`
- limit the client with client-id = 1234 to 1000 requests per 10 seconds
- all other clients will get the 403 status code

Given configuration:

```
{
  "routes": [
    {
      "sourcePath": "/items",
      "destinationUrl": "https://example.com/items"
    }
  ],
  "clientIds": [
    {
      "clientId": "1111",
      "limit": 0
    }
  ]
}
```

When the client-id header contains 1111 then status code 429 should always be returned as the limit is always reached and for all other clients will get the 403 status code.

Bonus tasks if there are time left:

- configured routes persisted and should be available once the service restarts.
- REST API extended with CRUD operations
    - create a new route
    - modify the selected route
    - delete the selected route

Please provide a github repository link once you have the implementation.