Code Assignment

Required deliverables

- Working example
- Access to the code
- Summary of design considerations
- Proposed next steps/improvements

Consider these aspects

- Scalability
- Performance
- Unit tests / Coverage

Constraints

Javascript / TypeScript

Use case

We're building an application to maintain a product catalog that offers a REST API.
We should be able to:

- List products
- Add new products
- Update products
- Remove products

Changes to the catalog need to be propagated to 3rd party downstream services (e.g. supply chain) via a REST interface.

The partner's supply chain APIs are unreliable, and can return unexpected random errors, hence our application needs to be fault tolerant.
In case the API is down or returns an error, the system should ensure the message is properly processed once it recovers.

Mock API

You can use this mock API:
https://ev5uwiczj6.execute-api.eu-central-1.amazonaws.com/test/supply-chain

The mock has a few limitations:

- It is multi-tenant so others could be doing inventory updates as well.
- The data is stored in memory and not guaranteed to be persisted
- There’s no authentication

This is the OpenAPI specification of the mock API:

```
openapi: "3.0.1"
info:
  title: 'Supply Chain API'
  version: "1"
servers:
- url: "https://ev5uwiczj6.execute-api.eu-central-1.amazonaws.com/{basePath}"
  variables:
    basePath:
      default: "test"
paths:
  /supply-chain:
    get:
      responses:
        "200":
          description: "200 response"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Products"
    post:
      requestBody:
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/Product"
        required: true
      responses:
        "201":
          description: "201 response"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Product"
  /supply-chain/{id}:
    get:
      parameters:
      - name: "id"
        in: "path"
        required: true
        schema:
          type: "string"
      responses:
        "200":
          description: "200 response"
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Product"
    delete:
      parameters:
      - name: "id"
        in: "path"
        required: true
        schema:
```

```
          type: "string"
      responses:
        "204":
          description: "204 - No content"
components:
  schemas:
    Products:
        title: "List of products"
        type: "array"
        items:
          $ref: "#/components/schemas/Product"
    Product:
      title: "Product Schema"
      type: "object"
      properties:
        id:
            description: Product identifier
            type: string
        name:
            description: Product name
            type: string
        price:
            description: Price
            type: number
        quantity:
            description: Quantity
            type: integer
```