Project Documentation

Project

Overview

This system consists of two main projects:

1. **State CSV Batch Processor**

   Reads a CSV file containing phone records, batches the data by state, and sends it to the aggregator service via HTTP.
2. **State Aggregator Service**

   Receives state population batches, aggregates and stores them in a database, and exposes endpoints for querying and managing the data.

---

1. State CSV Batch Processor

Location: c:\Users\Wictorsama\source\desafio\state-csv-batch-processor

Purpose

- Reads src/data/phone_data.csv containing phone records.
- Batches records by state and sends them to the aggregator service endpoint /states/batch.
- Provides logging for processed and skipped records.

Main File

- src/main.ts

Key Components

- **CsvReaderService:** Reads and streams the CSV file line by line.
- **Batching Logic:** Collects records into batches of 1000 (configurable via BATCH_SIZE). Each record is validated to ensure it has a state and name. Each valid record is counted as one person for the state.
- **HTTP Client:** Uses axios to POST batches to the aggregator service.

- **Logging:** Logs the number of processed and skipped records, as well as batch send status.

How It Works

1. Starts a NestJS app (for Swagger and possible future API endpoints).
2. Reads the CSV file.
3. For each valid record:
   - Adds to the current batch.
   - When the batch size is reached, sends the batch to the aggregator service.
4. After all records are read, sends any remaining records in a final batch.
5. Logs the total number of processed and skipped records.

Example CSV Record

```
id,name,phone,state
1,John Doe,(11) 99999-9999,SP
```

Example Batch Sent

```
{
  "states": [
    { "name": "SP", "population": 1 },
    { "name": "RJ", "population": 1 }
    // ...
  ]
}
```

---

2. State Aggregator Service

Location: c:\Users\Wictorsama\source\desafio\state-aggregator-service

Purpose

- Receives batches of state population data.
- Aggregates and stores the total population per state in a database (MongoDB).
- Exposes REST API endpoints for querying and managing state data.

Main Controller

- src\presentation\controllers\state.controller.ts

Key Endpoints

POST
/states/batch
Receives a batch of state records

GET
/states/aggregates
Returns aggregated population per state

GET
/states/count
Returns the count of state records

DELETE
/states/all
Deletes all state records

Method	Endpoint	Description

Example: POST /states/batch

- Request Body:

```
{
  "states": [
    { "name": "SP", "population": 1 },
    { "name": "RJ", "population": 1 }
  ]
}
```

- Response:

```
{
  "success": true,
  "processedStates": 2,
  "aggregatedData": [
    { "state": "SP", "population": 1 },
    { "state": "RJ", "population": 1 }
  ]
```

```
}
 ``` Example: GET /states/aggregates
```

- Response:

```
[
  { "state": "SP", "totalPopulation": 1234 },
  { "state": "RJ", "totalPopulation": 567 }
]
```

```
```

Database

- Uses MongoDB to store state population data.
- Each document represents a state and its total population.

---

3. Running the Projects

Prerequisites

- Node.js and npm installed
- MongoDB and Redis running (can use Docker Compose)

Steps

1. Start MongoDB and Redis (if needed):

```
docker-compose up -d mongodb redis
```

2. Start the Aggregator Service:

```
cd c:\Users\Wictorsama\source\desafio\state-aggregator-service
npm install
npm run start:dev
```

```
```
```

3. Start the State CSV Batch Processor:

```
cd c:\Users\Wictorsama\source\desafio\state-csv-batch-processor
npm install
npm run start:dev
```

```

```

4. Access Swagger UI for API testing:
    - http://localhost:3001/api (if running via Docker Compose, otherwise check your port)

4. Troubleshooting

- **No data in /states/aggregates:**
    - Ensure both services are running.
    - Check logs for errors in batch processor and aggregator service.
    - Verify MongoDB contains documents in the states collection.
- **CSV not fully processed:**
    - Check logs for processed/skipped counts.
    - Ensure CSV file path is correct and file is accessible.

---

5. Extending the System

- You can adjust the batch size in src/main.ts (BATCH_SIZE).
- Add more validation or transformation logic in the batch processor.
- Add more endpoints or features in the aggregator service as needed.

All project names, paths, and references have been updated to reflect your new structure.