Usage-Estimate Pricing Service

Context and References

- **Problem Statement Document**: https://docs.google.com/document/d/122hWuyRH3_P4h2fPVVOY_o1ZzXpuiYYu-2fWpsNTy9Q/edit?tab=t.0

Notes and Assumptions

- The current implementation assumes the smallest block for time range is a **"Day"**
- For the current version the Pricing Service is also receiving "Days" as the time interval. In order to make it more generic and flexible, we can extend this and use **"Bucket Aggregation"** (e.g. using Mongo DB **Aggregation Framework**) to accumulate UserData, e.g. from days to monthly sum amount, when interacting with Pricing Service.
- The Pricing Service is mocked to return some hardcoded amounts in the current implementation.
- UsageData **attributes** have been modeled as a **generic** Map<String, String>, so any kind of parameter can be used in POST/GET API calls and stored in the Mongo Document DB.
- Caching and Pre-computing price estimates can be used to achieve better scalability and performance.
- The implementation shows the **minimum proof of concept** which works end to end. The details can change to model a more flexible and accurate UsageData / Pricing interaction.

Code Structure

- **com.continuous**
  - **pricing** (mock for external Pricing Service)
    - **controller**
      - **PricingController** (mock Service)
      - **PricingServiceDTO** (RequestBody)
      - **PricingServiceRestClient** (used as a proxy for REST API calls to Pricing Service)
  - **usageestimate**
    - **controller** (UsageEstimate RESTController / Service)
      - **UsageEstimateController** (Service implementation)
      - **UsageEstimateQuery** (instead of RequestBody, this can alternatively be passed as RequestParams)
    - **model** (Object/Entity model)
      - **PriceEstimate**
      - **PricingPlan**
      - **ResourceType**
      - **UsageData**
    - **repository** (for Document Store using MongoDB)
      - **PriceEstimateRepository**

- **UsageDataRepository**

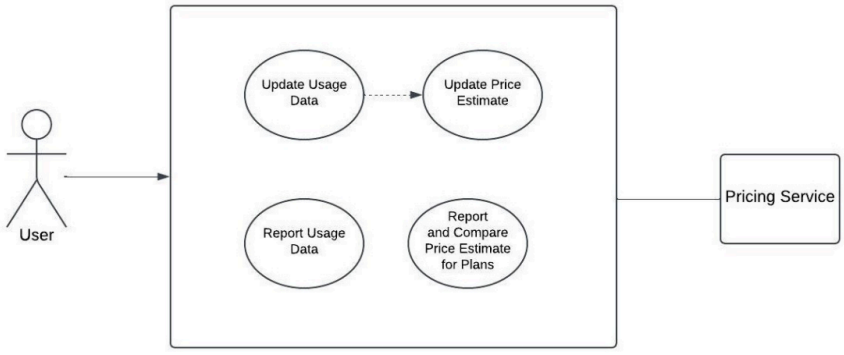
How to Run the application

- Run the following command in the project root directory (to bring up mongodb service):
  - `dokcer-compose up -d`
- Run the `Application` class in `com.continuous.usageestimate` package
- Some example API call (e.g. using Postman) are put at the end of this document

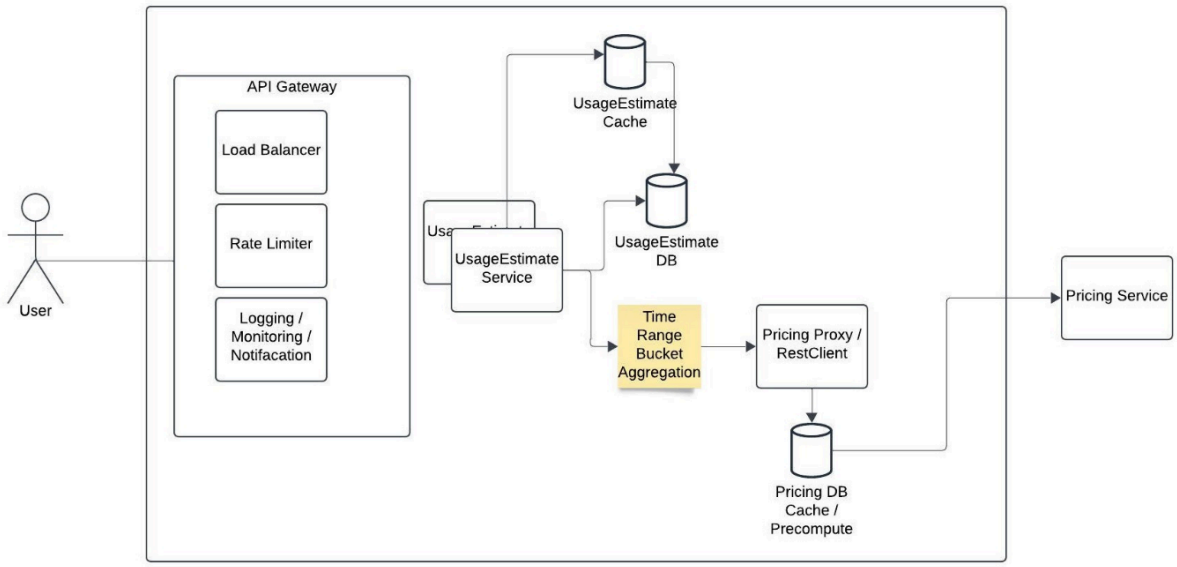
---

Appendix and Diagrams

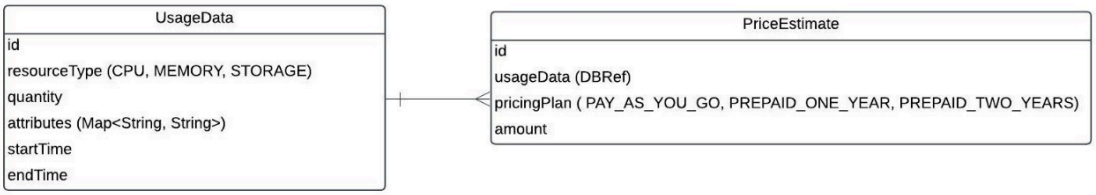
Context / Use Case Diagram



Architecture Diagram



Object Model

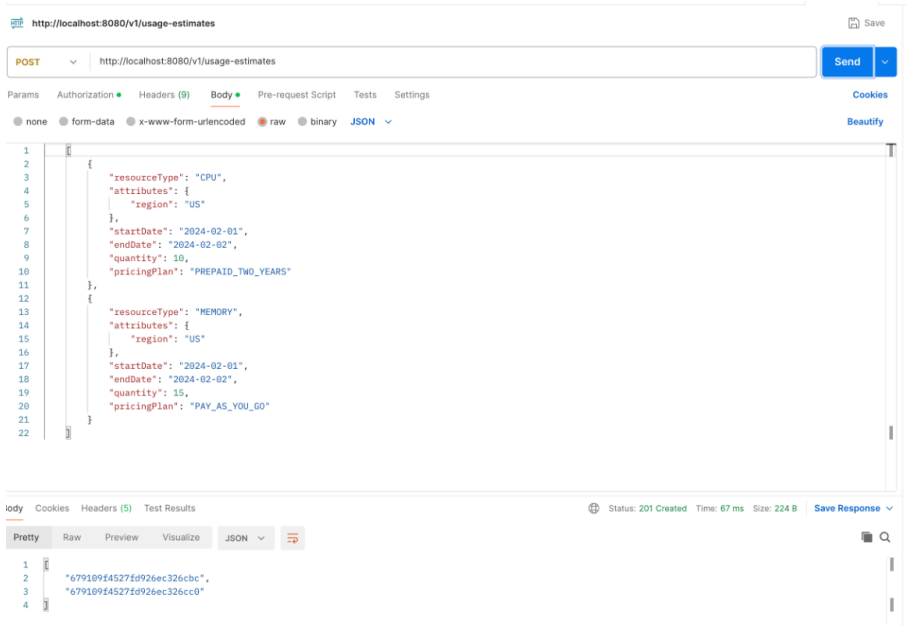


Usage-Estimates API

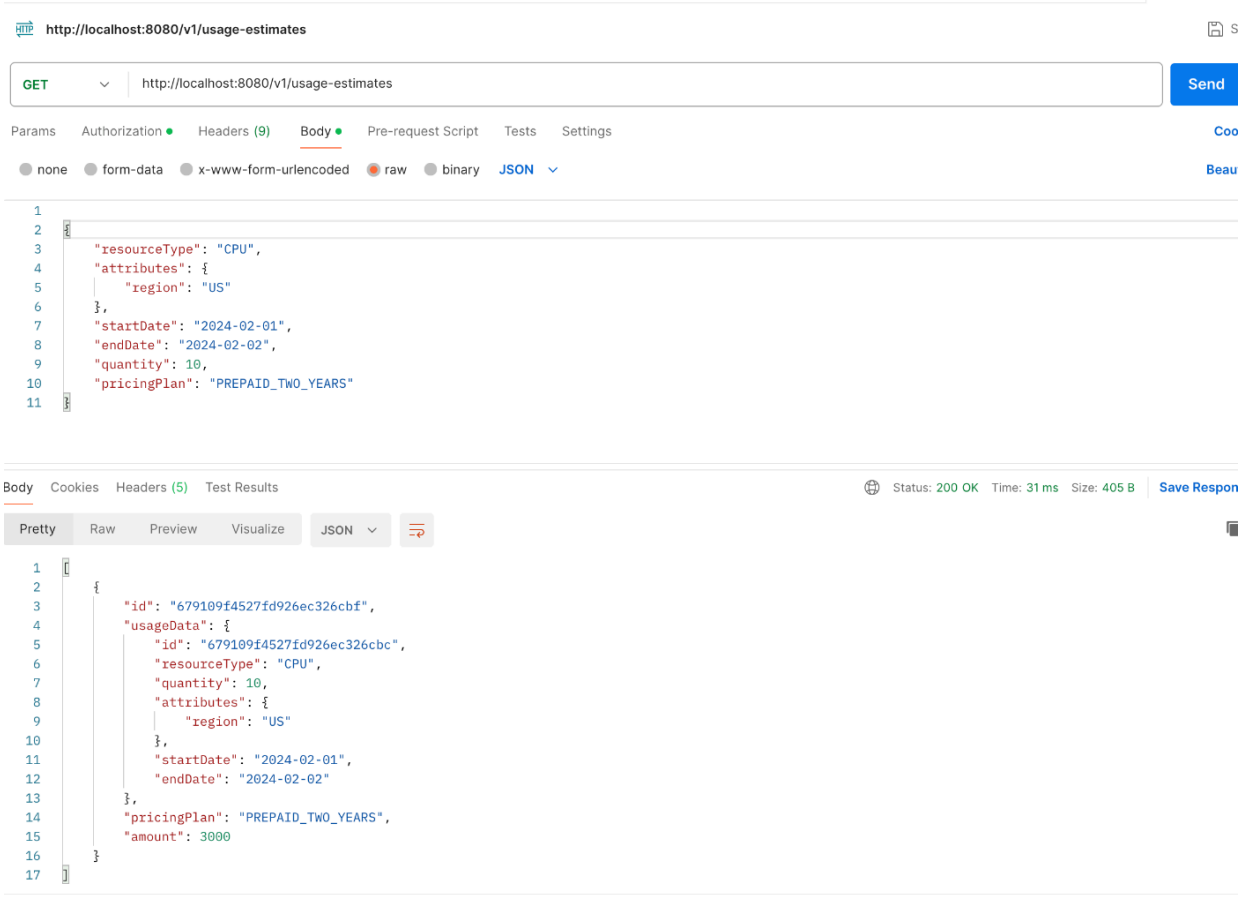
GET /v1/usage-estimates
- Retrieve usage data for a given resource, time period, and attribute set.
- Retrieve price estimates for a specific pricing plan, resource type, and time range.
- Compare price estimates across multiple pricing plans for the same usage data.

POST v1/usage-estimates
- Bulk Update of UsageData for multiple intervals and attributes.
- Cascading updates: When usage data is updated, all associated price estimates must be recalculated using the external pricing service.

Example POST Call (“Bulk” as an array)



Example GET Call


http://localhost:8080/v1/usage-estimates
GET
http://localhost:8080/v1/usage-estimates
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
JSON
{
"resourceType": "CPU",
"attributes": {
"region": "US"
},
"startDate": "2024-02-01",
"endDate": "2024-02-02",
"quantity": 10,
"pricingPlan": "PREPAID_TWO_YEARS"
}
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 31 ms
Size: 405 B
Pretty
Raw
Preview
Visualize
JSON
[
{
"id": "679109f4527fd926ec326cbf",
"usageData": {
"id": "679109f4527fd926ec326cbc",
"resourceType": "CPU",
"quantity": 10,
"attributes": {
"region": "US"
},
"startDate": "2024-02-01",
"endDate": "2024-02-02"
},
"pricingPlan": "PREPAID_TWO_YEARS",
"amount": 3000
}
]