**Grafana Dashboard Implementation Report**

**1. Data Source Setup**

* **Infinity Plugin (by yesoreyeram)** was installed and configured in Grafana Cloud.
* This allows CSV files to be used directly as a data source.
* Verified successfully via **Save & Test**.

**2. Dashboard Variables**

Two global variables were created for both dashboards:

1. **service**
   * Type: Custom
   * Values: All, Order Management, Account Management, Product Config
   * Include All enabled → allows wildcard filtering.
2. **status\_class**
   * Type: Custom
   * Values: 4xx, 5xx
   * Include All enabled → allows filtering for client/server errors.

👉 These variables are used across both dashboards and make drill-downs dynamic.

**3. App Services Health Dashboard**

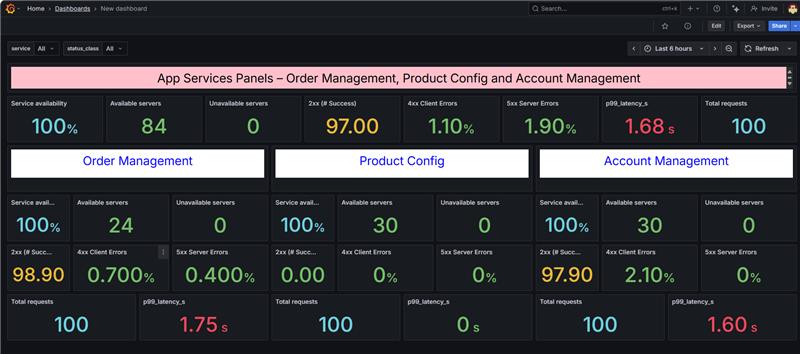
This dashboard provides an overview of **service health and performance metrics** across all services and per-service breakdowns.

**A. Global Summary Row (All Services)**

8 key metrics (using services\_summary.csv):

* Service availability (%)
* Available servers
* Unavailable servers
* Total requests
* 2xx Success Rate (%)
* 4xx Client Errors (%) → 🔗 linked to drill-down
* 5xx Server Errors (%) → 🔗 linked to drill-down
* P99 Latency (seconds)

✅ Each panel configured with:

* **Transformations** (Filter by service = All, select the right field).
* **Units** (%, count, seconds).
* **Thresholds/Colors** (e.g., latency >2s = red).
* **Data links** on error panels → wired to "Errors – Details".  
    
  

**B. Per-Service Sections**

For **Order Management, Product Config, and Account Management**, you duplicated the global summary panels and:

* Changed the filter to the respective service.
* Updated panel titles.
* Kept the same 8 metrics (Availability, Servers, Requests, 2xx/4xx/5xx, P99 Latency).
* Added drill-down links on **4xx and 5xx panels**, passing service + error type variables.  
    
  A screenshot of a computer

  AI-generated content may be incorrect.

**4. Errors – Details Dashboard**

This dashboard allows **deep dive into error codes and events**.

**A. Variables**

Reused the same **service** and **status\_class** variables.

**B. Panels**

1. **Errors by Code (Table)**
   * Source: errors\_by\_code.csv
   * Filters:
     + service = $service
     + status\_class = $status\_class
   * Group by: service, status\_class, status\_code
   * Aggregation: sum(count)
   * Shows error breakdown (e.g., 404 vs 500 vs 502).

A screenshot of a computer

AI-generated content may be incorrect.

1. **Recent Error Events (Table)**
   * Source: error\_events.csv
   * Filters: same as above.
   * Shows detailed logs: ts, service, status\_code, route, request\_id, message.

A screenshot of a computer

AI-generated content may be incorrect.

**5. Drill-Down Wiring**

* On the Health Dashboard → 4xx and 5xx panels → **Data Links** → linked to "Errors – Details".
* Variables passed:
  + service (either All or specific service).
  + status\_class (4xx or 5xx).
* ✅ Verified: Clicking on error rate panels takes you directly to the Errors dashboard, already filtered.

**6. Outcome**

* **App Health Dashboard** provides a **summary view** for availability, request success, errors, and latency.
* **Errors – Details Dashboard** enables a **drill-down** into specific error codes and events.
* Together, they form a **multi-level observability workflow**:
  + **Top row** = global SLA view.
  + **Per-service rows** = SLO/SLA at component level.
  + **Error drill-downs** = root cause exploration.
* Dashboards are **interactive**, respond to **time range filters**, and support **service/error filtering** via variables.  
  