Visualisation Dashboard for BRM (BSP)

We will be using Grafana, which is an open-source data visualization and monitoring platform that allows users to create, explore, and share dashboards and visualizations based on data from various sources. It provides a user-friendly interface for creating and managing dashboards that can display data in the form of graphs, tables, alerts, and more.

As I completed the Time Series Database Design earlier, the next part was to connect the grafana with InfluxDB using FluxQL for data visualisation. Although the queries need to be optimized more and need to be written as per the user's perspective (Here, Data Engineers from BSP).

For now, I have written a script with will stress our database designed in InfluxDB with random() values, and a basic dashboard to test out if the has been visualized correctly or not.

Below are the options we have as visualisations in grafana:

- Graphs & charts
    - Time series is the default and main Graph visualization.
    - State timeline for state changes over time.
    - Status history for periodic state over time.
    - Bar chart shows any categorical data.
    - Histogram calculates and shows value distribution in a bar chart.
    - Heatmap visualizes data in two dimensions, used typically for the magnitude of a phenomenon.
    - Pie chart is typically used where proportionality is important.
    - Candlestick is typically for financial data where the focus is price/data movement.
- Stats & numbers
    - Stat for big stats and optional sparkline.
    - Bar gauge is a horizontal or vertical bar gauge.
- Misc
    - Table is the main and only table visualization.
    - Logs is the main visualization for logs.
    - Node Graph for directed graphs or networks.
    - Traces is the main visualization for traces.
    - Flame Graph is the main visualization for profiling.
- Widgets
    - Dashboard list can list dashboards.
    - Alert list can list alerts.
    - Text panel can show markdown and html.
    - News panel can show RSS feeds.

Adding FluxQL queries for visualisation:

Flux is a query language used in InfluxDB, and it is also supported in Grafana. Here are the steps to design a FluxQL query for Grafana:

1. Open the Grafana UI and navigate to the dashboard you want to create a Flux query for.
2. Click on the "Add Query" button to open the query editor.
3. In the query editor, select "Flux" as the query language.
4. Write your Flux query in the editor. The syntax for FluxQL is similar to SQL, but with some differences. For example, instead of using "SELECT" to select data, you use "from", and instead of using "WHERE" to filter data, you use "|> filter()" or "|> range()".
5. Once you have written your query, click on the "Run Query" button to test it. You should see the results of your query displayed in the preview pane.
6. If your query is working correctly, click on the "Apply" button to add the query to your dashboard.
7. You can then use the visualizations available in Grafana to display the results of your Flux query.

Here's a sample of a FluxQL query that retrieves data from an InfluxDB database:

```
from(bucket: "snsrs_data")
  |> range(start: v.timeRangeStart, stop: v.timeRangeStop)
  |> filter(fn: (r) => r["_measurement"] == "data")
  |> filter(fn: (r) => r["_field"] == "snsr_value")
   |> filter(fn: (r) => r["snsrs_id"] == "2LyHplR7" or r["snsrs_id"] == "3OYqxVTh" or r["snsrs_id"] == "4hHnZiK3" or r["snsrs_id"] == "65Oa2I1l")
```

This query retrieves data from a bucket “snsrs_data”, filters the data to include only measurements with a tag called "_measurement" that equals "data".

To display dashboard to end-user (On a secondary screen or on a big screen/TV)

We will be using Grafana Kiosk, it is a feature in Grafana that allows users to display dashboards in a full-screen mode without the Grafana UI elements (such as the navigation bar and panels) visible. This is useful for creating dashboards that can be displayed on a TV or monitor in a public space, like an operations centre or a conference room.

When a dashboard is put into Kiosk mode, it takes up the full screen, and there is no way to access the Grafana UI. This prevents users from accidentally navigating away from the dashboard or modifying its configuration.

Also, a kiosk account which is a separate account with no admin access and ‘view’ only permission will be used for kiosk mode.

```
./bin/grafana-kiosk -URL=https://ip-addrs-of-server:port -login-method=gcom -username=kiosk_brm_bsp -password=abc:123 -kiosk-mode=full
```

References:

https://grafana.com/docs/grafana/latest/panels-visualizations/visualizations/

https://github.com/grafana/grafana-kiosk