**DevOps KPI**

**What is DevOps KPI?**

Key Performance Indicators are metrics widely used to know how good (or bad) are some practices, products, projects or even initiatives. Well, planning projects define KPIs that are collected from day zero and followed in all steps.

When a company starts some project with a focused goal it’s necessary to measure how effective and if the steps to take the target are correctly done. With KPI we should get the facts about our decisions, supporting then and know how good is the rolling work.

**Introduction**

As a culture, a transformation, a mindset change, new tools, highly automated process, and organizational changes, the DevOps initiative must have KPIs to measure it’s a success, because there are goals that must be reached and strategic decisions to take.

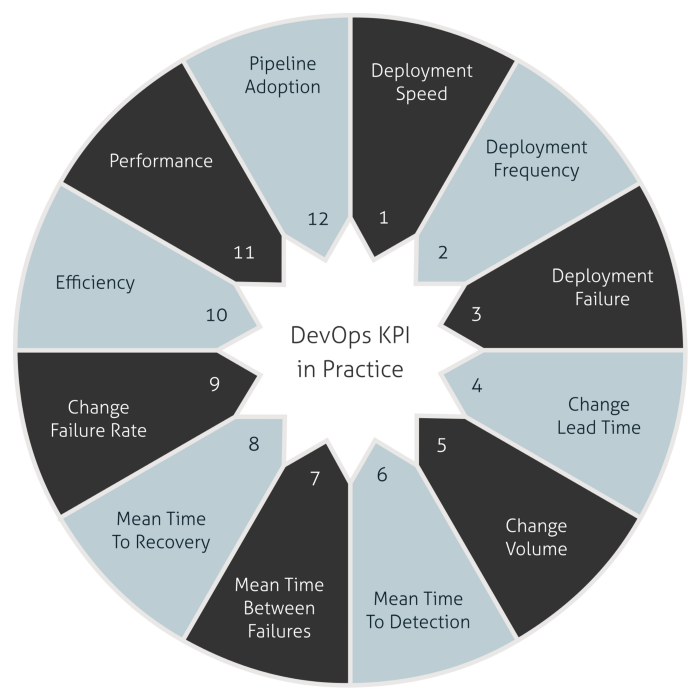
In this class, I will show up KPIs in practice, how to collect and process the metrics from the toolset that composes the DevOps pipeline, and how to use them to compute meaningful metrics.

**Example**

When company develops and launches a new product they need to know how good the product is. The KPI for this is how many orders they received to manufacture the new product, that can be called as Order Volume. But, for more concrete values they should use the number of invoices and name this as Invoice Volume.

With these numbers is easy to know the success: high numbers are good because it means the acceptance of this new product. And low numbers are bad and decisions may take the place, supported by KPI.

**Type of KPI’s**



**Deployment Speed:** knows how speed is the deployments thought of the pipeline.

**Deployment Frequency:** how many deployments happen in production.

**Deployment Failure:** how many deployments are failing in production environment.

**Change Lead Time:** the average time between developing a feature and to deploy it in production.

**Change Volume:** total new user stories and new lines of code that are shipped in each deployment in production.

**Mean Time To Detection:** the average time between deployment and identify the very first failure in a production environment.

**Mean Time Between Failures:** the average time between failures in production.

**Mean Time To Recovery:** the average time between a crash down of environment and recovery in production.

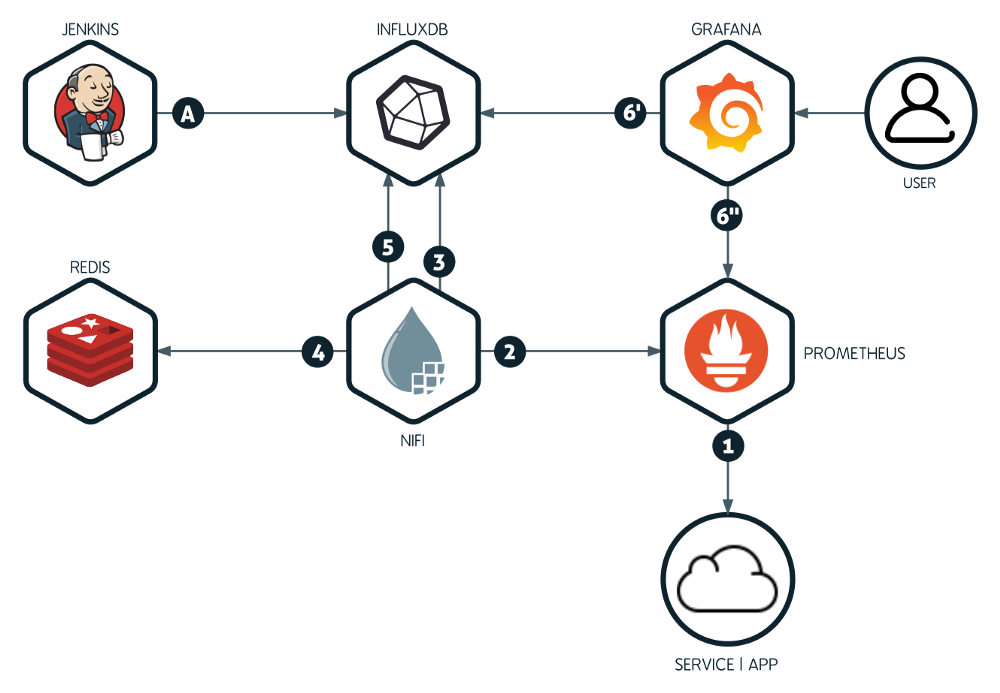
**Change Failure Rate:** the relation between changes and outages in production.

**Efficiency:** the equivalence between touch time and wait time. These two metrics are related to time to develop a feature and time waiting until deploying it in production.

**Performance:** this KPI shows the availability, response time and resource utilization of application in production.

**Pipeline Adoption:** understand how is the adoption of each definition you made to become a user of the brand new DevOps pipeline.

**Implementation of Deployment speed, frequency and Failure**



**Pushing the metrics:** Jenkins collects and sends the metrics to InfluxDB.

**Writing the joined metrics:** write again the joined data into InfluxDB.

**Reading the metrics:** Grafana reads and presents the metrics as KPIs.

**Accessing the metrics:** the Users observes the KPIs and take decisions.

**Jenkins:** acts as our metrics producer, pushing all collected data from every job execution to InfluxDB.

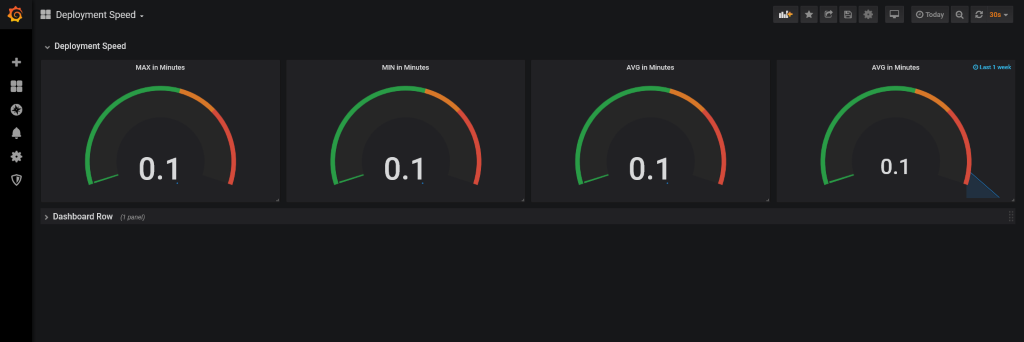
**InfluxDB:** acts as our metric repository and query engine, receiving all metrics and making it available to be presented by Grafana.

**Apache Nifi:** acts as our layer for the metrics normalization, transformation and in the next chapters, as metrics correlation.

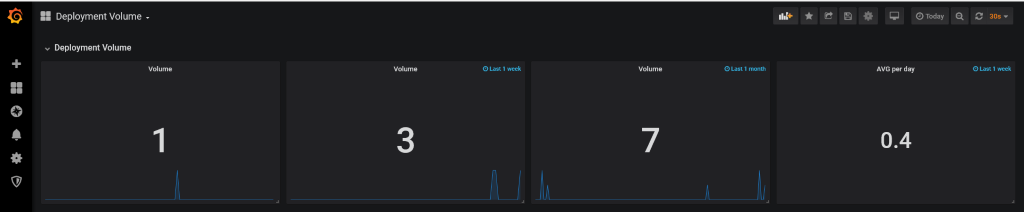
**Redis:** acts as our cache layer to store the latest metric timestamp to prevent double reads.

**Grafana:** it’s our metric presentation tool, with rich graphics and a lot of possibilities to create dashboards to show our KPIs.

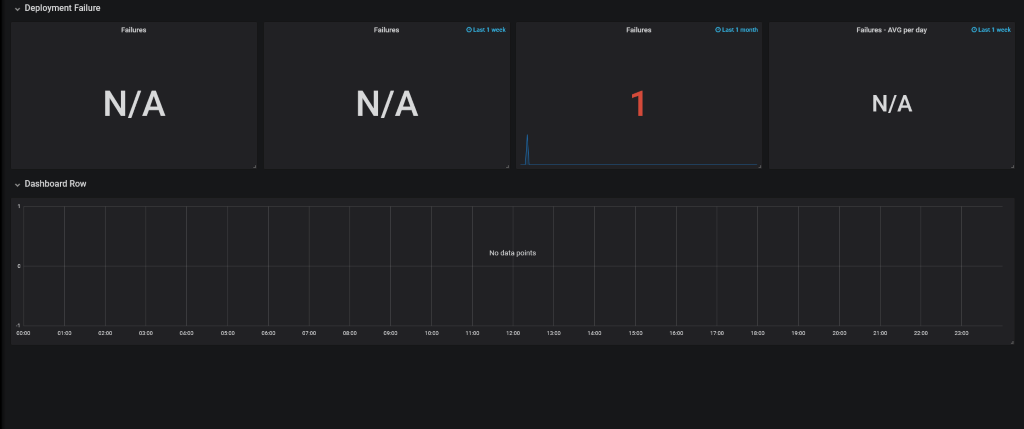
**Deployment Speed** — How fast is my pipeline to deploy in production?



**Deployment Volume** — What is my capacity to perform multiples deployments in a timeframe? A day for example.

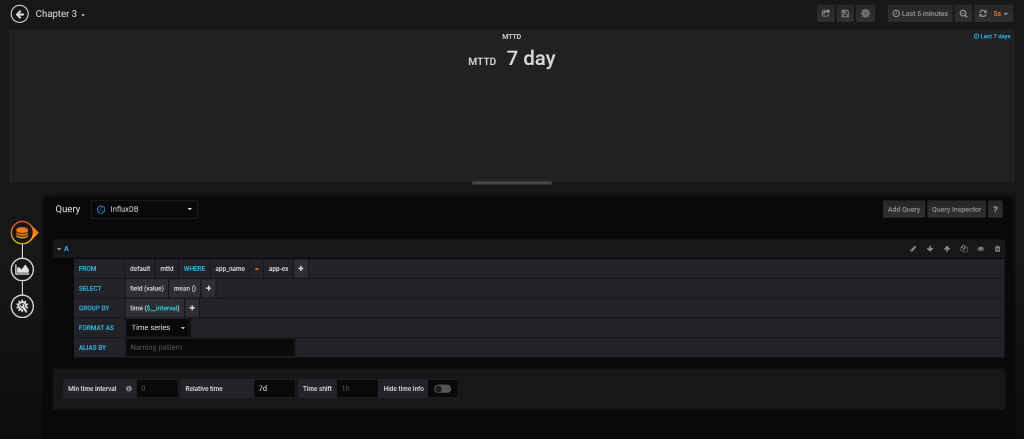


**Deployment Failure** — How accurate is my pipeline to deploy in production?

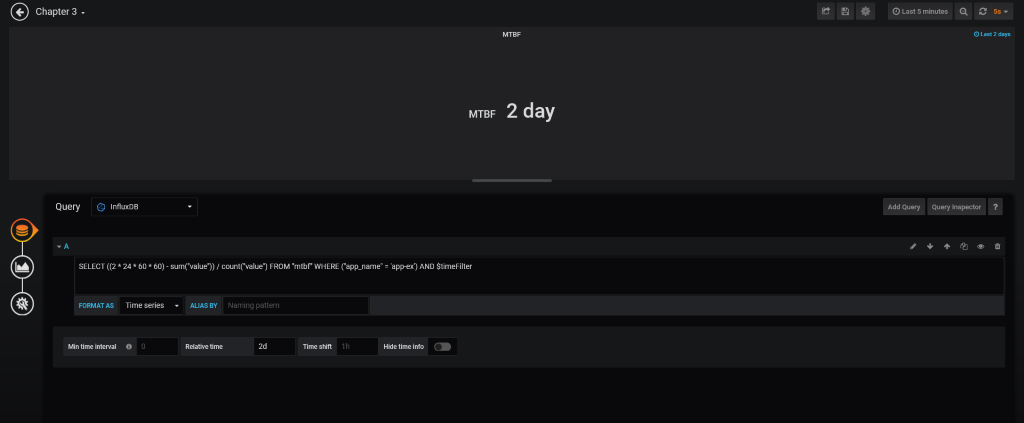


**MTTD and MTBF:**

**Mean Time to Detection:** how much time we run without failures in production?



**Mean Time Between Failures:** how much time we run gracefully until the next failure? how much our services are reliable in production?

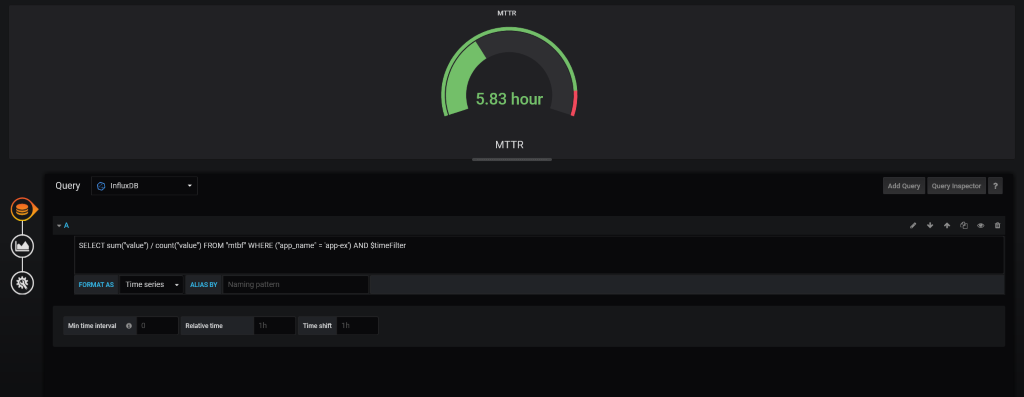


**Execution**

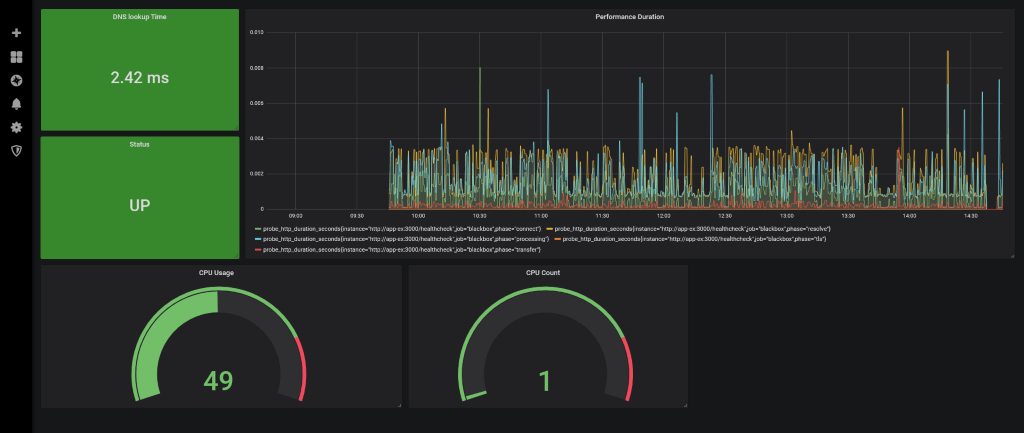
* Prometheus, through the Blackbox exporter, monitors the HTTP API. When it stops to respond, Prometheus starts to produce down metrics.
* Apache Nifi is listening for down metrics. When it happens, it starts our data pipeline to compute the KPIs.
* Nifi gets the deployment metrics to use in the MTTD computation.
* Redis plays an important role because we need a way to store a checkpoint to prevent double MTTD computation and to mark a downtime begin and a downtime end, for MTBF computation.
* Apache Nifi, after ending the KPI computation, pushes them to InfluxDB.
* Grafana fetches the data points from InfluxDB and shows them as beautiful charts. It’s using the data directly from Prometheus too, to show when our application is up or when is down.

**Mean Time To Recovery:** how much time I take to get up again.

Mean time to recovery is calculated by adding up all the downtime in a specific period and dividing it by the number of incidents.

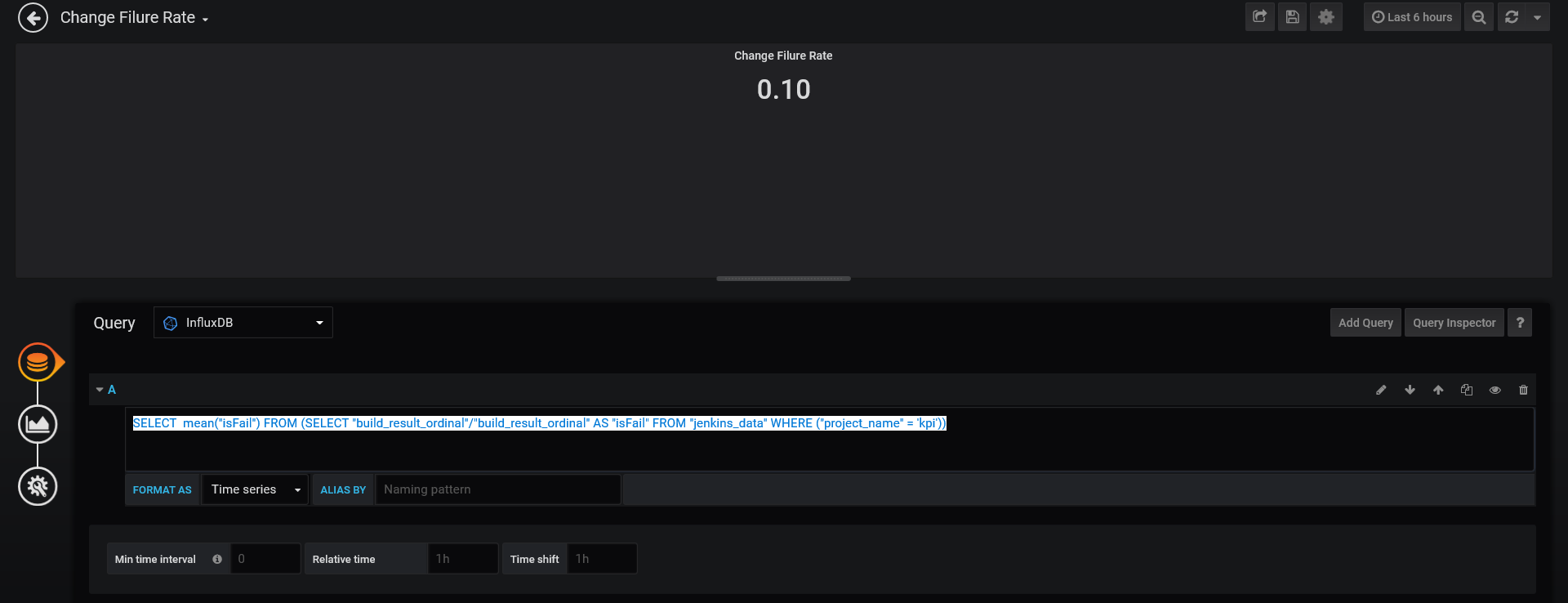


**Performance:** the performance of our running app it’s so important.



**Change Failure Rate:** what’s the relationship between new features and the failures in production.

Change failure rate is the number of deployments in which something goes wrong, out of the total number of deployments in any given period of time.



**GIT URL**

https://github.com/anilrh03/DevOps-KPI.git