Monitoring of real-time applications

Openhack - 2020

Contributors

Vikas Chahal (Mentor)
Rishabh Gupta (CSE, IIT Bhubaneswar)
Mounika Mukkamalla (CSE, IIT Indore)

Problem statement

- There are millions of requests to Unacademy services every minute.
- To manually check traffic, resource utilisation and other parameters for any system to ensure it's smooth functioning are an essential but a gigantic task.
- We need a service to track our systems and services' health. It has to monitor if the parameters of the system are within the threshold.

Project Objectives

01 To monitor health of existing services

02 The monitoring service should have reconfigurable conditions to be able to be used on system.

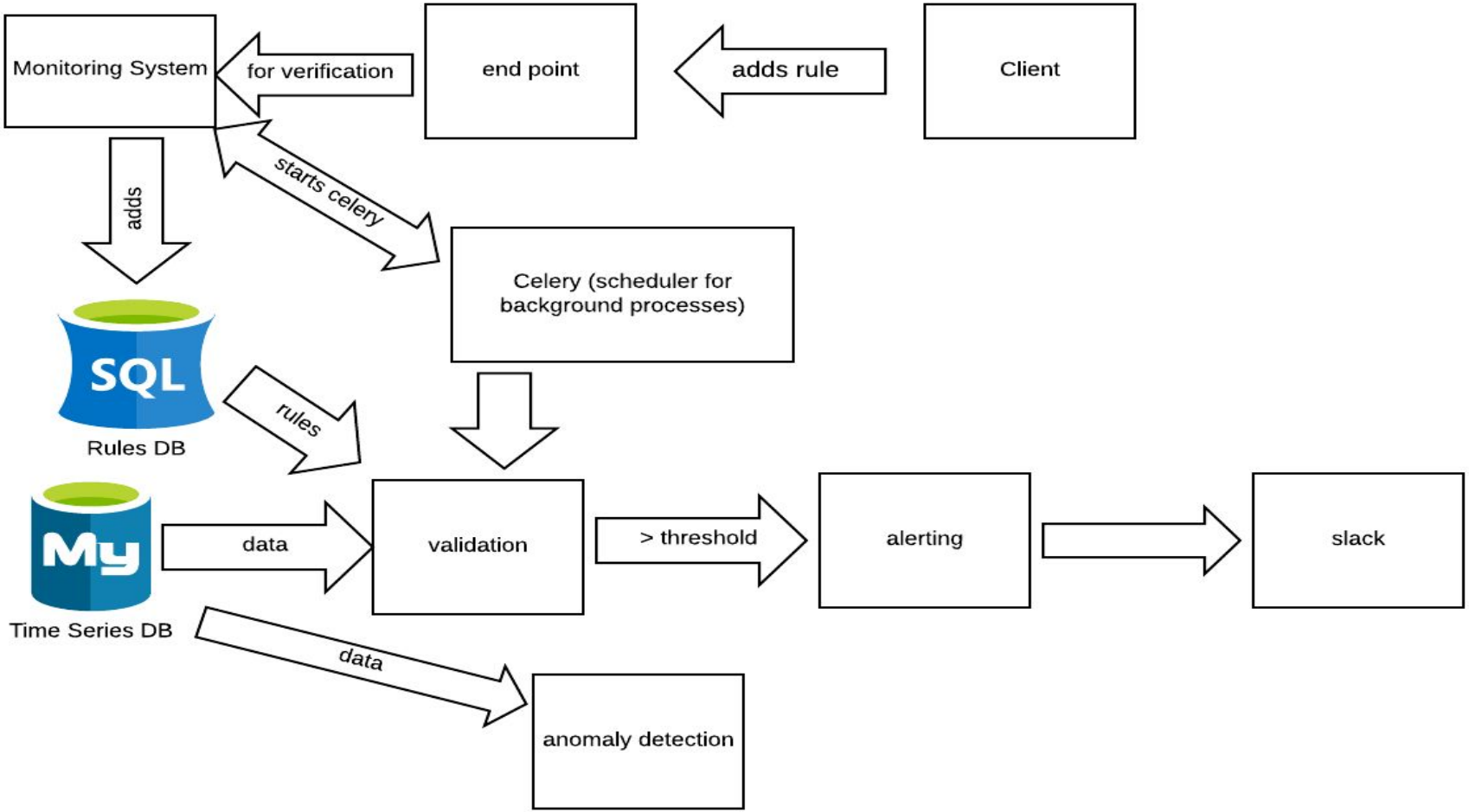
03 Smartly observe and detect anomaly to warn about potential attack or reconfiguration of thresholds. (additional feature)

Features

- This product is intended to boost the productivity of entire engineering team. Eg: DDOS attack
- Teams no longer have to manually check every system for and deduce if something is wrong in it.
- The small warnings often go unnoticed and crop up as bigger issues to solve later

Tech Stack

- Django (MVC web development framework)
- Time series DB (the data where conditions are checked)
- MySQL
- Celery (scheduling tasks)
- Grafana (MySQL query visualisation tool)
- ADTK Machine Learning Tools
- Redis (Broker)
- Slack Webhooks (for alerting)
- **A good leader and mentor: Vikas Chahal**


Monitoring System
for verification
end point
adds rule
Client
adds
starts celery
Celery (scheduler for background processes)
SQL
Rules DB
rules
My
Time Series DB
data
validation
> threshold
alerting
slack
data
anomaly detection

So How to use it?

- You add rules. Let's say P99 or avg of response time of requests > Threshold send an alert to slack.
- We validate this rule. Rule can be very complicated with multiple conditions in it. Like: (a>b && c>d)&&(c+f>e-g). If it's correct we add it to DB.
- Now these response time (or particular resource according to rule) will be pulled out to from Time Series DB and rules will be applied to it. If condition is true then an action like send alert message is performed.
- Now on all data, we run anomaly test to check for any aberrations and we mark them and send an alert.

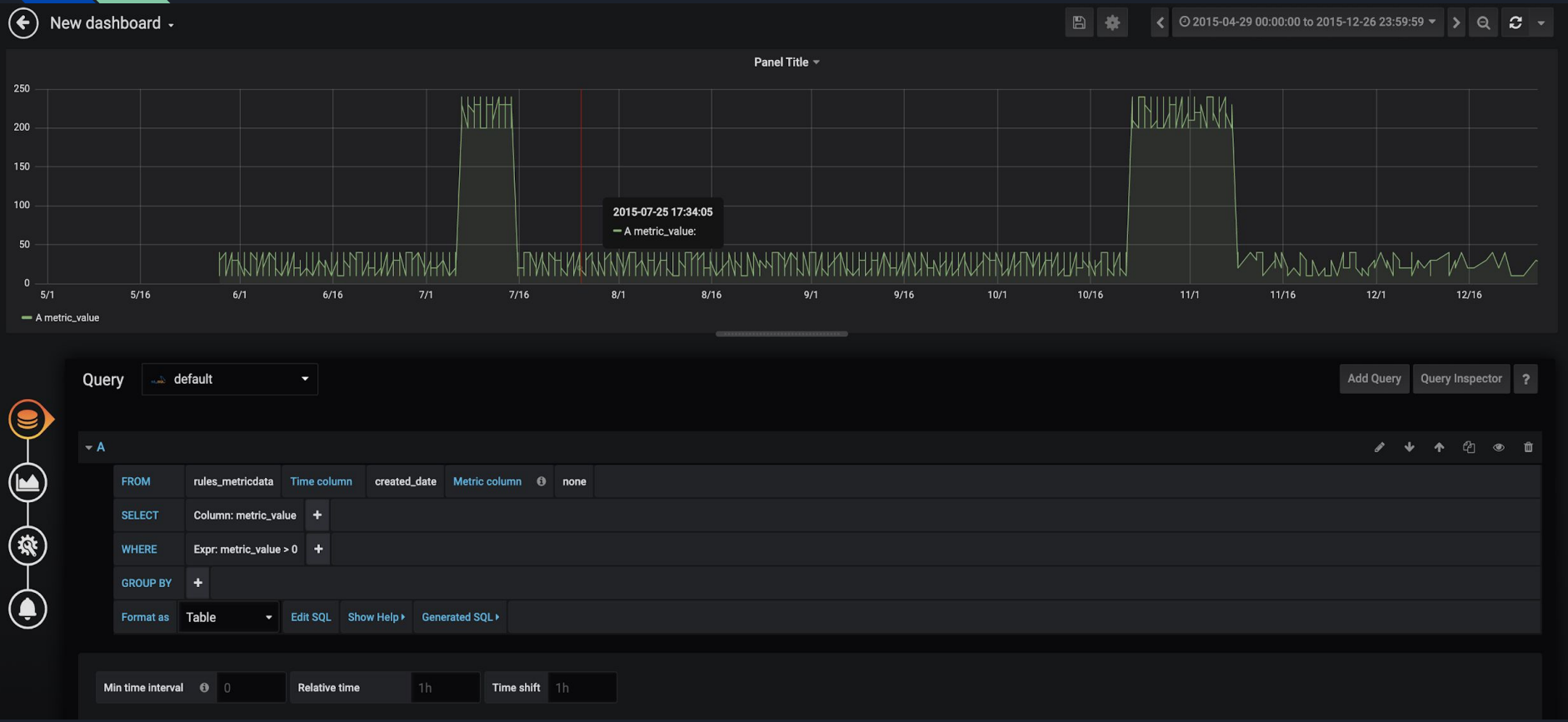
Objectives achieved

- A fully running service which is capable of monitoring every other service and system.
- Rules can be configured and any database can used for extracting data
- Additional plugin which was not in our earlier schedule, anomaly detection system is also developed.

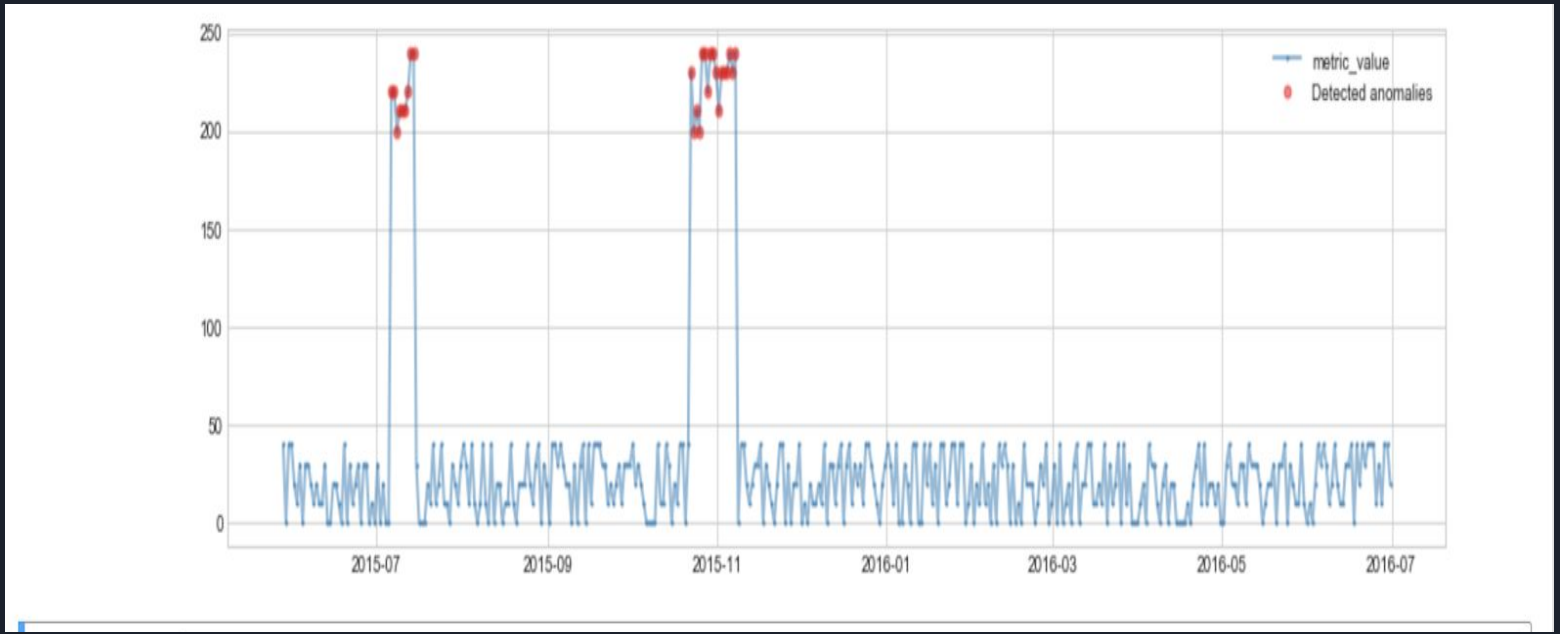
Outcomes

- Earlier the if the service parameters are within safe limits were checked by engineers here themselves but now this service will do it.
- Since this was a lengthy process and there was no general tool for all of them, this process was usually skipped. This means that it was more of “we will solve the problem as it comes approach”
- I think this will serve as necessary step to stop that.
- Also it is impossible for human mind to process millions of traffic parameters to check for potential and current anomalies. This tool will do that.
- I think it’s objective complete

Response time vs date



Anomaly Detector



Thank you!