**Cloud based Architecture Proposal**

* Continuous Integration (CI) and Continuous Delivery (CD) form the backbone of the product delivery lifecycle. A well tuned, fault tolerant and scalable CI/CD pipeline is very important to support modern Agile teams.
* Despite obvious business advantages, a rapid release approach combined with continuous change processes resulting from DevOps principles will in the long run generate new challenges. The entire process needs to be carefully examined and controlled. Neglecting this aspect can cost you dearly.
* A way to mitigate this risk is to implement a Continuous Monitoring and Observability solution—the final capability for a CI/CD pipeline. By invoking the DevOps “measure everything” principle, a Continuous Monitoring tool will allow you to take advantage of the following features:

**Analyzing long-term trends:**

* How many builds am I running daily?
* How many did I run a month ago?
* Do I need to scale my infrastructure up or down?

**Over-time comparison:**

* Is my deployment slower than it was last week?
* Or is it faster?

**Vulnerability scans:**

* Does my code introduce any critical software faults and security vulnerabilities such as memory leaks, uninitialized variables, array-boundary?
* How rapidly are they detected? How quickly are they fixed?

**Alerting:**

* Is something broken, or might break soon?
* Did my test pipeline pass?
* Should I perform a rollback on my latest deployment?

**The possible solution I would recommend for Cloud based architecture are:**

Data Store - Prometheus

Data visulaization – Grafana

**Prometheus (Data Store)**

* Prometheus is an open-source systems monitoring and alerting toolkit originally built at Sound Cloud.Prometheus scrapes metrics from jobs, either directly or via an intermediary push gateway

**Features**

1. A multi-dimensional data model with time series data identified by metric name and key/value pairs

2. PromQL, a flexible query language to leverage this dimensional

3. Time series collection happens via a pull model over HTTP

4. Pushing time series is supported via an intermediary gateway

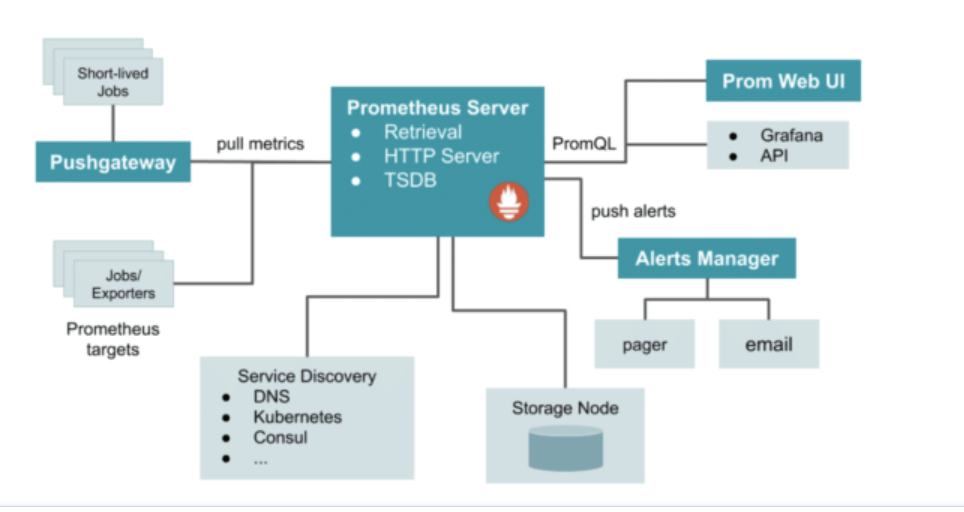
5. Targets are discovered via service discovery or static configuration

6. multiple modes of graphing and dashboard support

**Advantages of Prometheus over other monitoring tools**

* Prometheus is TSDB
* Prometheus is Pull based tool
* Centralised control
* In built Alerting facility
* Easy for monitoring teams
* Data visualisation
* Service discovery (sd)
* Scalability
* PromQL

ARCHITECTURE :



* Prometheus can scrape metrics from jobs directly or, for short-lived jobs by using a push gateway when the job exits. The scraped samples are stored locally and rules are applied to the data to aggregate and generate new time series from existing data or generate alerts based on user-defined triggers. While Prometheus comes with a functional Web dashboard or other API consumers can be used to visualize the collected data, with Grafana being the de facto default.

**How Prometheus works**

* Prometheus gets metric from an exposed HTTP endpoint. A number of client libraries are available to provide this application integration when building software. With an available endpoint, Prometheus can scrape numerical data and store it as a time series in a local time series database. It can also integrate with remote storage options.
* In addition to the stored time series, impermanent times series from the source are produced by queries. These series are recognized by metric name and key value pairs known by labels. Queries are generated using PromQL (Prometheus Query Language) that enables users to choose and aggregate time series data in real time. PromQL is also used to establish alert conditions that can then transmit notifications outside sources such as PagerDuty, Slack or email. These data can be displayed in graph or tabular form in Prometheus’s Web UI. Alternatively, and commonly, API integrations with alternative display solutions such as Grafana may be used.

**Is Prometheus the Correct Monitoring Solution?**

* Prometheus’ primary focus is on reliability rather than accuracy. For this reason, it is ideal in highly dynamic systems such as microservices running in a cloud environment. It is probably not a good fit for a system that requires high accuracy, such as a billing application. In this case the specific billing function should be addressed with an alternative, but Prometheus may still be the right tool for monitoring the other application and infrastructure functions.
* The focus on reliability is built in by making each Prometheus server standalone with local time series database storage to avoid reliance on any remote service. This design makes Prometheus an ideal tool to rely on for rapidly identifying issues and getting real-time feedback on system performance.

**Grafana ( Data visualization)**

* Grafana allows you to query, visualize, alert on and understand your metrics no matter where they are stored.
* Create, explore, and share dashboards with your team.
* Grafana is written in go language.

**Features and Benefits**

* Even small businesses generate vast quantities of data in 2021, making tools for tracking and analyzing big data an essential part of any workflow. Grafana is an effective solution because it is easy to use, provides beautiful graphs, and can be integrated with a number of databases, including MySQL, Graphite, Influx DB, Logz.io, Influx DB, ElasticSearch, and PostgreSQL. The fact that it is open source also means that developers can create custom plugins to meet their specific needs.
* Grafana is an excellent tool for visualizing data. Accessibility and observability for all users are ensured by streamlined installation and configuration. All users can find tracking application performance a satisfying experience thanks to the attractive and customizable interface.
* The dashboard is well-equipped to make sense of complicated data like graphs, heatmaps, histograms, and geo maps, and it is constantly upgraded. The tool provides a variety of visualization solutions for understanding data in line with the business needs of a specific project. Here are the key features of Grafana that you should know:

**Grafana benefits include:**

* Customizable dashboards are feature-rich and can be configured to display data from a wide range of databases using visualization tools such as heatmaps, histograms, and charts. The platform is flexible and easy to use.
* Native support of a broad range of databases. Grafana is open source and provides the ability to create custom plugins based on project requirements.
* Grafana is the ideal solution for dashboard development and visualization because cloud systems like AWS have built-in data collection infrastructure.
* You can host it on-premises or on any cloud platform you choose.

**Is Grafana the Correct Monitoring Solution?**

* Companies that use database analytics and visualization tools like Grafana are much more efficient than their competitors. Grafana is used by businesses to track their infrastructure and log analytics and improve operational performance. Grafana dashboards facilitate the monitoring of users and events by automating the collection, management, and display of data. This information can be used by product managers, security analysts, and developers to guide their decisions.
* It reveals to teams and businesses what their customers really do, not just what they claim to do. These are known to be revealed behaviors, and they are extremely informative. Users aren’t very good at foreseeing their own future, but having analytics helps tech teams to dig deeper than human-error-prone surveys and tracking.

I have attached the architecture diagram below:

PROPOSED ARCHITECTURE:

