Tarea Corta I - Observability

Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
Bases de Datos II (IC 4302)
Segundo Semestre 2022

1. **Objetivo General**
   - ➔ Implementar una solución de Observability para bases de datos SQL y NoSQL.

2. **Objetivos Específicos**
   - ➔ Instalar y configurar motores de bases de datos SQL y NoSQL mediante Kubernetes.
   - ➔ Instalar y configurar una solución de monitoreo y alertas utilizando Prometheus y Grafana.
   - ➔ Implementar pruebas de carga sobre bases de datos SQL y NoSQL mediante el uso de la herramienta Gatling.

3. **Datos Generales**
   - ➔ El valor de la tarea corta: 7%
   - ➔ La tarea debe ser implementada por grupos máximo de 5 personas.
   - ➔ La **fecha de entrega** es 09/09/2022 antes de las 11:30 pm.
   - ➔ Cualquier indicio de copia será calificado con una nota de 0 y será procesado de acuerdo con el reglamento. La copia incluye código que se puede encontrar en Internet y que sea utilizado parcial o totalmente sin el debido reconocimiento al autor.
   - ➔ La revisión es realizada por el profesor asignado al curso, él mismo se reserva el derecho de solicitar una revisión virtual con los miembros del grupo para evacuar cualquier duda sobre la implementación.
   - ➔ Se espera que todos y todas las integrantes del grupo entiendan la implementación suministrada.
   - ➔ Se deben seguir buenas prácticas de programación. Por ejemplo, documentación interna y externa, estándares de código, diagramas de arquitectura, diagramas de flujo, pruebas unitarias son algunas de las buenas prácticas que se esperan de un estudiante de Ingeniería en Computación.
   - ➔ El lenguaje de implementación en caso de ser requerido debe ser seleccionado por cada grupo y debe estar debidamente documentado.
   - ➔ Toda documentación debe ser implementada en Markdown.
   - ➔ El email de entrega debe contener una copia del proyecto en formato tar.gz y un enlace al repositorio dónde se encuentra almacenado, debe seguir los lineamientos en el programa de curso.
   - ➔ Al no entregar documentación se obtiene una nota de 0.

4. **Descripción**

Debido a la explosión en el uso de microservicios en los años recientes, el desarrollo de aplicaciones más interactivas y tecnologías como Internet of Things, se han comenzado a generar grandes cantidades de datos de monitoreo, entre las cuales podemos encontrar métricas de sistemas, logs y traces, esta situación limita la posibilidad de que seres humanos y sistemas computacionales convencionales (bases de datos SQL) puedan extraer información valiosa que permita entre otras cosas entender y predecir el comportamiento del sistema con el fin de tomar las medidas pertinentes a tiempo para asegurar su máximo disponibilidad, evitando actuar de forma reactiva.

Con la generación de grandes cantidades de datos de series temporales, se han desarrollado bases de datos NoSQL llamadas Time Series Databases, una de las cuales es Prometheus, la cual ha evolucionado y se ha convertido en la solución OpenSource líder para Monitoreo, la misma se integra con otras herramientas OpenSource como Kubernetes, Grafana y Thanos para formar un suite de aplicaciones que pueden competir con las soluciones SaaS lideres en el mercado como lo son Datadog, NewRelic y Dynatrace.

Para este trabajo, cada uno de los grupos deberá llevar a cabo las siguientes tareas para implementar una solución de monitoreo:

Instalación de motores de base de datos

Para el desarrollo de esta tarea cada uno de los grupos deberá elaborar Helm Charts que permitan la instalación y configuración de las siguientes herramientas:

- Prometheus
  - https://bitnami.com/stack/prometheus-operator/helm
- Grafana
  - https://bitnami.com/stack/grafana/helm
- MariaDB (versión OpenSource de MySQL)
  - https://bitnami.com/stack/mariadb/helm
  - mínimo de 3 instancias con un primary y dos replicas.
- MongoDB
  - https://bitnami.com/stack/mongodb/helm
  - mínimo de 3 replicas
- Elasticsearch
  - https://www.elastic.co/guide/en/cloud-on-k8s/2.2/index.html
  - mínimo de 3 data nodes con un máster node
- PostgreSQL
  - https://bitnami.com/stack/postgresql/helm

Se recomienda la elaboración de los siguientes Helm Charts:

- **databases**: Este contendrá las siguientes herramientas
    - MariaDB
    - MongoDB
    - Elasticsearch
    - PostgreSQL

    El uso del motor de bases de datos deseado se puede controlar mediante dependencias y condiciones a nivel de Helm Charts, por ejemplo, después de ejecutar ***helm create databases***, se puede agregar el siguiente código al archivo **Chart.yaml**

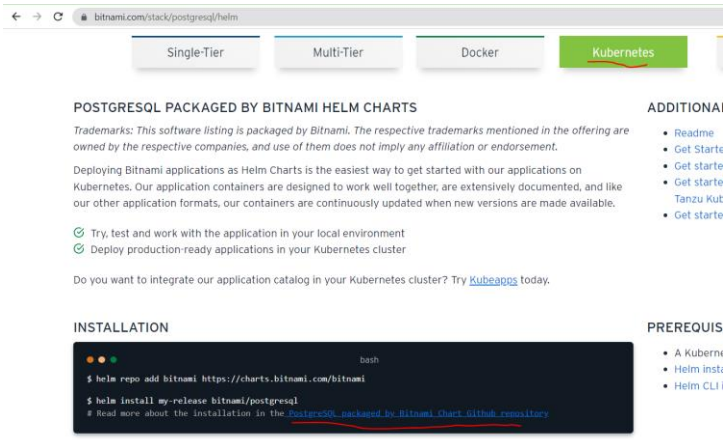
```yaml
dependencies:
  - name: eck-operator
    version: "2.3.0"
    repository: https://helm.elastic.co
    condition: enableECKOperator
  - name: mariadb
    version: "11.1.8"
    repository: https://charts.bitnami.com/bitnami
    condition: enableMariaDB
  - name: kube-prometheus
    version: "8.0.16"
    repository: https://charts.bitnami.com/bitnami
    condition: enablePrometheus
```

    Y luego agregar en el archivo **values.yaml** lo siguiente:

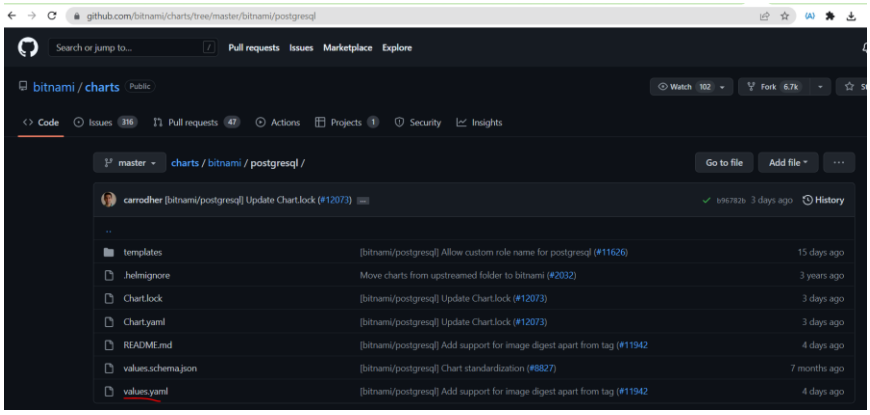
```yaml
enableMariaDB: true
enablePrometheus: true
enableGrafana: true
enableECKOperator: false
enableElasticsearch: true
enableElasticsearchExporter: true
```

- monitoring: Contendrá las herramientas Prometheus, Grafana y Thanos en caso de que se desee usar.

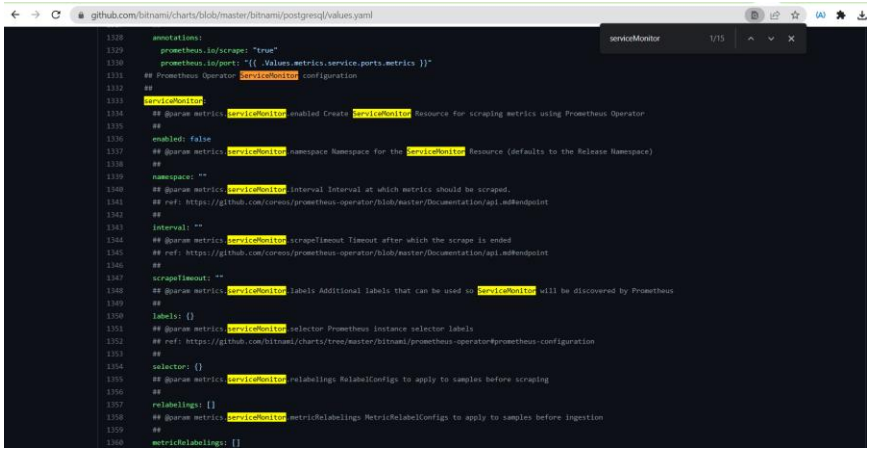
Es importante mencionar que cada uno de los paquetes de instalación proporcionados por Bitnami, se encuentran preparados para generar métricas de uso consumidas por Prometheus de forma dinámica (auto-discovery) para habilitarlo basta con usar los parámetros correctos para el helm chart, por ejemplo en el caso de PostgreSQL, en la documentación y en la sección de instalación, se le debe dar clic al link *"# Read more about the installation in the PostgreSQL packaged by Bitnami Chart Github repository"*



Esto los llevara al repositorio de GitHub de este Helm Chart, https://github.com/bitnami/charts/tree/master/bitnami/postgresql, en el mismo se encuentra un archivo llamado values.yaml



En este archivo se encuentra todas las posibles configuraciones para el Helm Chart, basta con buscar *serviceMonitor,* para encontrar como habilitar la integración con Prometheus.

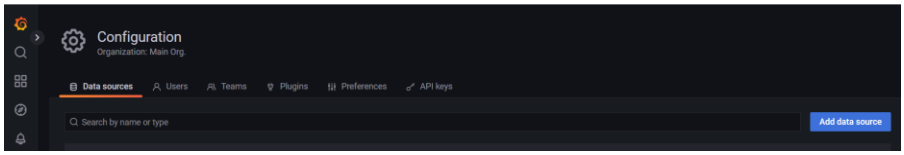


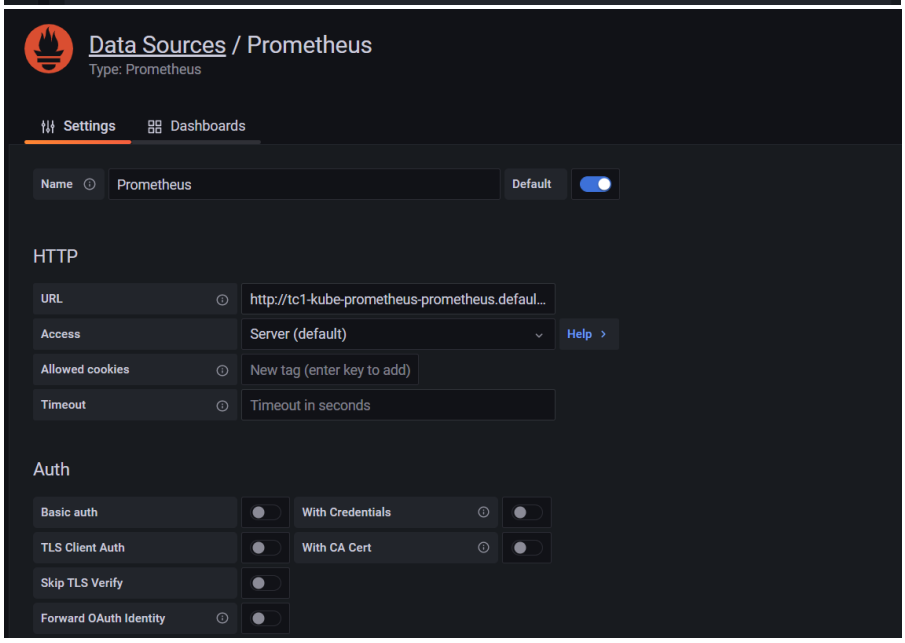
En el caso de Elasticsearch, los paquetes oficiales de ECK no contienen una integración con Prometheus, por esta razón es necesario la siguiente herramienta https://prometheus-community.github.io/helm-charts/

Configuración de Grafana

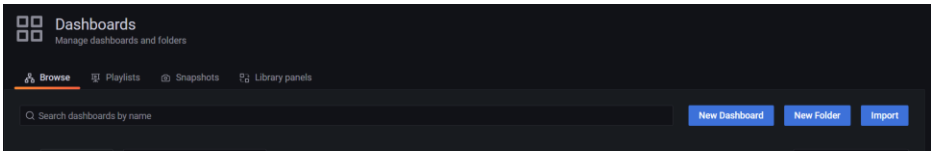
Una vez que se tienen las configuraciones de las bases de datos y el sistema de monitoreo listas, se debe configurar Grafana, basta con redireccionar el servicio e ingresar con un browser, se deben de llevar a cabo las siguientes acciones:

- Agregar un datasource el cual debe ser Prometheus.





- Agregar Dashboards para cada uno de los servicios que se desean instalar, existen gran cantidad de Dashboards disponibles en el sitio web de Grafana https://grafana.com/grafana/dashboards/



Por ejemplo, luego de agregar el de MySQL Overview, el resultado es:

Generación de pruebas

Cada uno de los grupos, deberá generar pruebas de carga con Gatling, las mismas realizaran en los motores de bases de datos los siguientes tipos de pruebas:

- Creación de registros/documentos.
- Borrado de registros/documentos.
- Actualización de registros/documentos.
- Búsquedas de registros/documentos.

Las mismas deberán correr por largos periodos de tiempo (al menos 30 minutos), para permitir generar gráficos de monitoreo donde se pueda observar el comportamiento de los diferentes motores de bases de datos, es importante observar los siguientes parámetros (no todas las bases de datos los exponen):

- Disco.
- Memoria.
- CPU.
- Network.
- File Descriptors.
- IOPS.
- Open Connections.
- Queries per second.
- Query response time.
- Thread Pools

La idea detrás de las pruebas de carga es medir el rendimiento de las bases de datos con una configuración especifica.

Documentación

La documentación debe incluir al menos:

- Guía de instalación y uso de la tarea: Se debe explicar en detalle como ejecutarla y como configurar componentes no automatizados (por ejemplo, configuración de Grafana).
- Configuración de las herramientas (haciendo énfasis en los valores utilizados para cada una).
- Pruebas de carga realizadas: se debe especificar el tipo de datos que se están almacenando (dataset), tipo de prueba (creación, borrado, actualización y búsqueda), parámetros utilizados (configuración de Gatling), resultados (apoyados por la información de monitoreo y gráficos obtenidos) y conclusiones de la prueba, se deben incluir al menos 5 pruebas por motor de búsqueda
- Conclusiones y recomendaciones de la tarea corta.

6. **Entregables**
   - Documentación.
   - Proyecto del Gatling con todos los scripts requeridos.
   - Helm Charts para la instalación de las herramientas requeridas por la tarea.

7. **Evaluación**

Funcionalidad / Requerimiento	Porcentaje
Helm Charts	50%
Pruebas Gatling	25%
Documentación	25%