

Proyecto 2 Journal Search Platform (JSP)

Instituto Tecnológico de Costa Rica
Escuela de Ingeniería en Computación
IC-4302 Bases de Datos II
Prof. Nereo Campos
IIS 2022

Integrantes

Johnny Díaz Coto	- 2020042119
Pamela González López	- 2019390545
Andrea Li Hernández	- 2021028783
Deyan Sanabria Fallas	- 2021046131
Andrés Sánchez Rojas	- 2018100180

Instrucciones de como ejecutar el proyecto

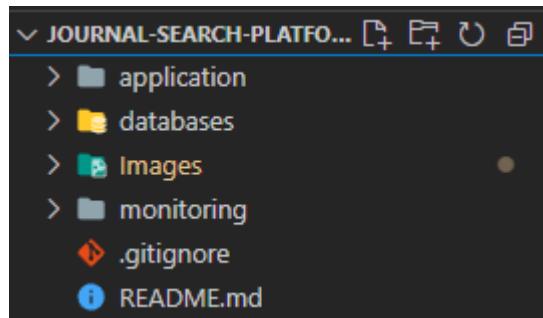
El proyecto se encuentra en un repositorio privado de GitHub, el profesor del curso o las personas permitidas pueden realizar la instalación sin ningún problema, en el caso de personas fuera del repositorio, en caso de cambiarse la configuración de este también podrían hacer las mismas cosas que se explicaran a continuación.

Clonación del Repositorio

Para poder clonar los documentos que se encuentran en GitHub se debe realizar el siguiente comando:

```
git clone https://github.com/jonkox/Journal-Search-Platform-JSP.git
```

Esto lo que hará es descargar el proyecto dentro de la computadora que lo ejecute. Es recomendable tener una cuenta de GitHub para evitar algunos problemas con la autenticación. Hecho este paso debe obtener el proyecto con una distribución así:



El proyecto se constituye de varias carpetas, cada una de ellas tiene su función, sin embargo se pueden dividir en dos grupos, estos son:

- Helm Charts
 - application
 - databases

- monitoring
- Aplicaciones
 - Images
 - API
 - Details Downloader
 - Downloader
 - Jatsxml Processor
 - Loader
 - Pruebas

Las aplicaciones contienen todo el código necesario para ejecutar la tarea que cada una de ellas debe de cumplir, estas se dividen en el código de producción y la rama de pruebas. Por otro lado, los Helm Charts son utilizados para instalar todas las herramientas dentro de Kubernetes, donde cada Helm Chart instala distintas herramientas de la siguiente manera:

- application
 - Instala todas las aplicaciones creadas por el equipo de trabajo.
- databases
 - Instala las bases de datos que se necesitan en el proyecto, ElasticSearch y MariaDB y RabbitMQ.
- monitoring
 - Instala todas las herramientas de monitoreo del proyecto: prometheus y grafana.

Pasos de Instalación

1. Posicionarse en la carpeta principal del proyecto.
2. Ejecutar el comando:

```
helm install [name][chart]
```

En nuestro caso serían los siguientes comandos:

```
helm install databases databases
helm install monitoring monitoring
helm install application application
```

Nota

Al crear un Helm Chart e instalarlo, se va a utilizar el nombre del chart seguido por el nombre del componente en cuestión, *[nombre del chart]-[nombre del componente]*. En nuestro caso por ejemplo, con el Helm Chart encargado del monitoreo tiene como nombre “monitoring”, así que un ejemplo práctico del nombre de un pod es **monitoring-mariadb**, dónde esto lo podemos saber con anticipación gracias al nombre del chart y el componente que se desea instalar. Ahora, esto nos afecta directamente en distintas situaciones, dónde continuando con el ejemplo de monitoring:

- En el momento de obtener la contraseña de MariaDB por medio de variables de entorno se tiene la siguiente sección dentro del deployment: ``yaml
- name: "MARIADBPASS" valueFrom: secretKeyRef: name: databases-mariadb key: mariadb-root-password optional: false

Al momento de escribir el código necesario para esta tarea se tiene en cuenta los nombres de los componentes para así conocer desde antes su valor, por tanto, es importante tener en cuenta el nombre de los charts al momento de su creación y programación a sus al rededores.

Ejecución del proyecto

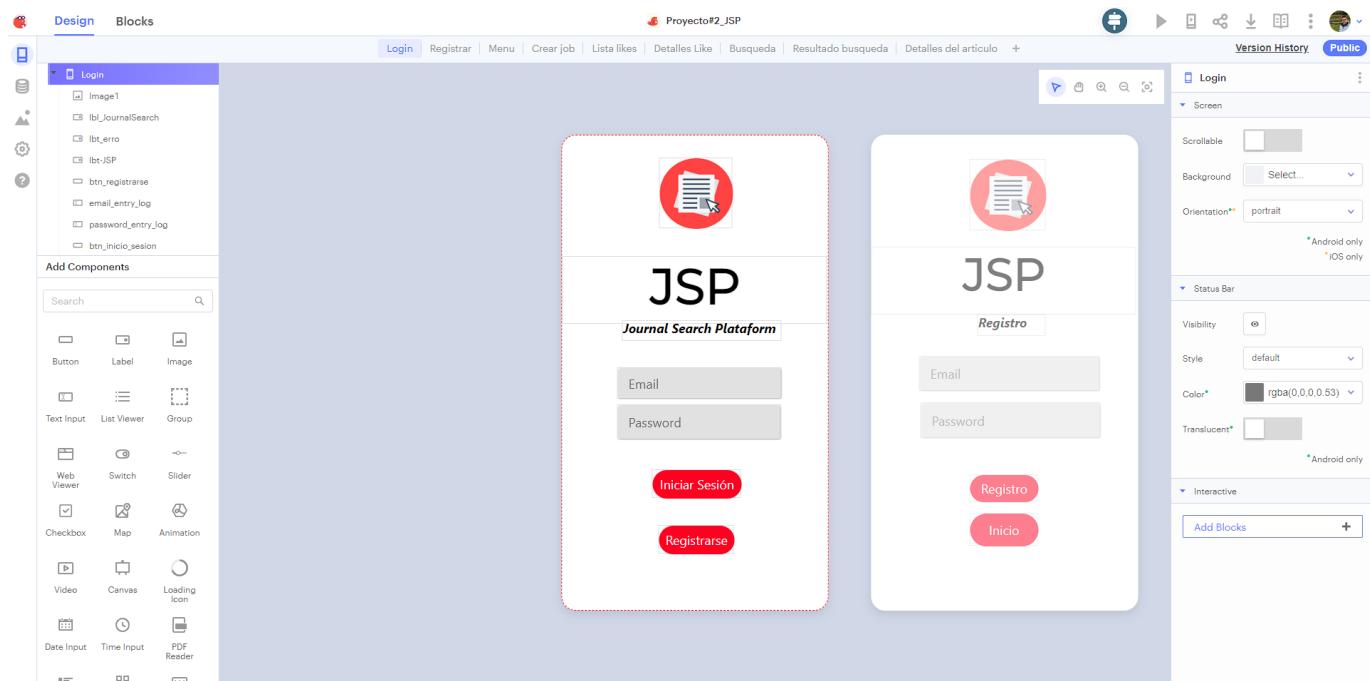
Thunkable

Para esta aplicación se debe estar registrado previamente en Thunkable. Usted puede crear una copia ([link](#)) con la cual puede trabajar. Una vez con su copia puede ejecutar el proyecto desde su computador o desde un dispositivo móvil como un celular o tablet.

- Versión Web De esta manera, una vez en el proyecto en la sección superior derecha se puede encontrar la siguiente barra de herramientas.



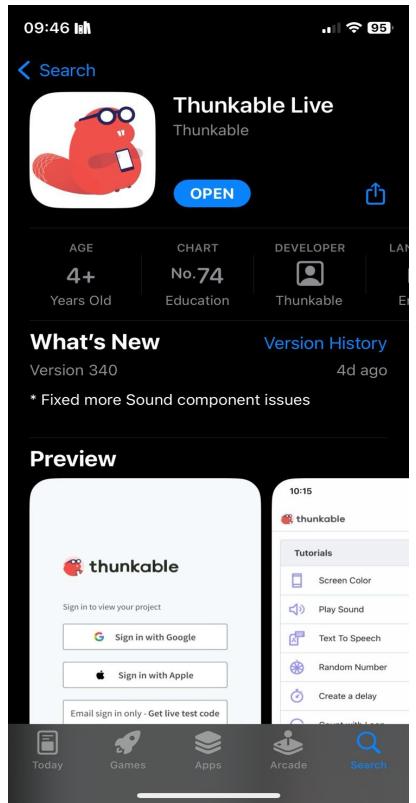
con ella se pueden realizar las distintas acciones para ejecutar la aplicación, el componente de más a la izquierda con un dibujo de `play` consta de la prueba de una versión web de la aplicación, la cual mostrara algo similar a la siguiente imagen:



De este modo podrá realizar las distintas opciones que la aplicación posee.

- Versión Móvil Como se puede observar en la imagen anterior, la misma página oficial de recomienda el uso de la aplicación móvil, pues, es una aplicación móvil, lo mejor sería probarla en un dispositivo móvil como tal. Para ello se necesita tener instalado en el dispositivo la aplicación de Thunkable.

Journarl Search Platform

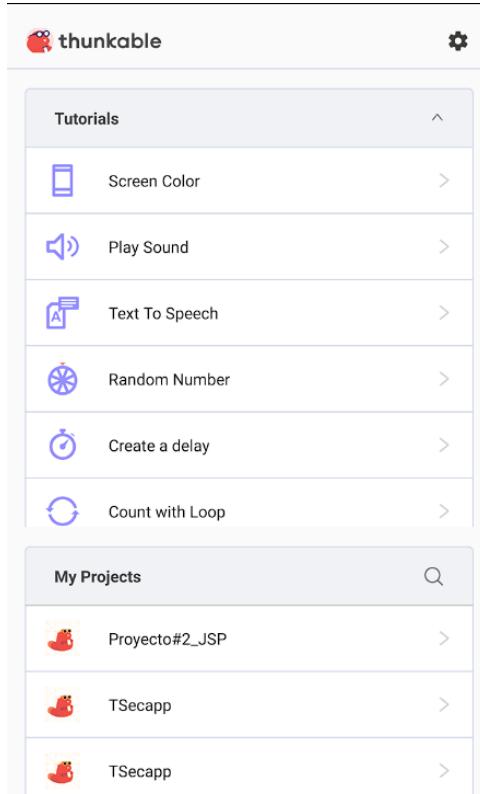


Para lograr la prueba móvil se debe seleccionar la opción de *live test on device* y entrar a la aplicación de Thunkable Live en el dispositivo móvil.

Thunkable live

En la web de Thukable después de haber seleccionado la opción de *live test on device* debe iniciar su aplicación de Thunkable Live donde vera algo similar a lo siguiente:

Journarl Search Platform



Debe seleccionar su proyecto, en este caso es el proyecto de Proyecto#2_JSP. A partir de este punto el usuario se encuentra en libertad de probar cada una de las funcionalidades de la app.

Ngrok

Para utilizar esta herramienta se debe descargar dentro del computador, esto se puede realizar de una forma muy sencilla en caso de tener las herramientas necesarias para su instalación se puede usar los comandos:

Windows

```
choco install ngrok
```

Mac

```
brew install ngrok/ngrok/ngrok
```

En caso de no contar con las de Chocolatey o Hombrew de igual manera se puede descargar el en un formato comprimido para su instalador.

Una vez terminada su descarga, dentro de la página oficial de [ngrok](#) se debe crear una cuenta, seguidamente debe obtener el [token](#) de autenticación para sincronizarlo en su aplicación local, para ello se ejecuta el comando:

```
ngrok config add-authtoken <token>
```

Journarl Search Platform

Donde el <token> es distinto de cada usuario. Una vez realizado este paso, ngrok está listo para funcionar y esto se logra con el siguiente comando:

```
ngrok http <port>
```

Donde <port> se refiere al puerto donde se creará el túnel para la comunicación entre las dos partes, la aplicación de Thunkable y el API. (por defecto es 5000)

Abrir puertos del API

Antes de ejecutar ngrok o la app, se necesita ejecutar el siguiente comando en una terminal:

```
kubectl port-forward SERVICE/api-service <puerto-deseado>:5000
```

Donde <puerto-deseado> es el puerto donde se hosteará localmente el API, este puerto es necesario para el siguiente paso.

Ejecutar Ngrok

Una vez expuesto el API, con la herramienta de Ngrok se va a exponer al API a "el resto del mundo", lo que va a generar un túnel entre el API y el Internet. Para lograr esto de buena manera debe tener en cuenta el puerto elegido en el paso anterior, en este ejemplo, el puerto es 63456, por lo que se ejecuta el siguiente comando:

```
ngrok [protocolo] [puerto]
```

Tomando en cuenta este ejemplo en concreto, el comando que se necesita ejecutar es el siguiente:

```
ngrok http 63456
```

Debe obtener un resultado parecido al siguiente:

```
ngrok                                     (Ctrl+C to quit)

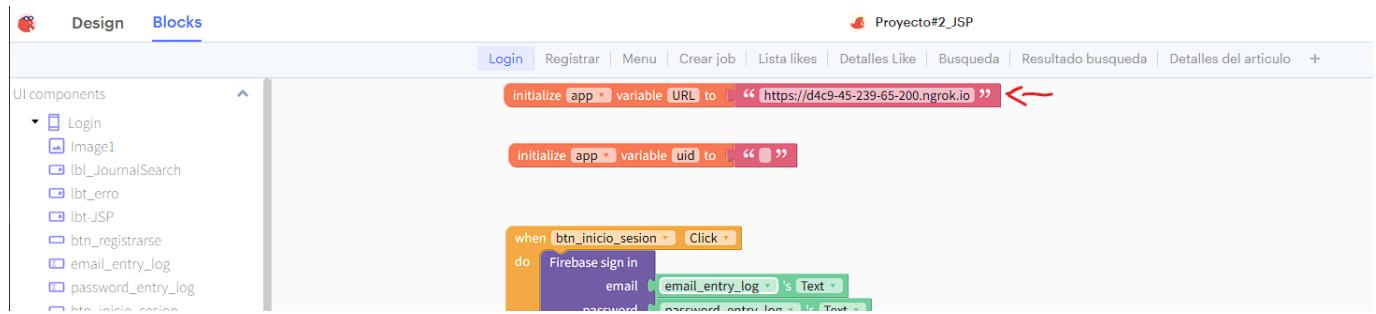
Try our new native Go library: https://github.com/ngrok/ngrok-go

Session Status      online
Account            Johnny Díaz (Plan: Free)
Version             3.1.0
Region              United States (us)
Latency             94ms
Web Interface      http://127.0.0.1:4040
Forwarding          https://3d66-186-179-67-163.ngrok.io -> http://localhost:5000

Connections        ttl     opn     rt1     rt5     p50     p90
                    0       0       0.00    0.00    0.00    0.00
```

De nuevo, note el recuadro amarillo, pues este indica el URL al cual puede acceder al API de forma pública, así de esta manera la app se puede comunicar con el API.

Thunkable



En esa sección hay que colocar el URL obtenido anteriormente en Ngrok.

Una vez realizado todo lo anterior, solo queda utilizar la APP, para un ejemplo, ir a las [pruebas generales](#)

Explicación de Secciones Varias

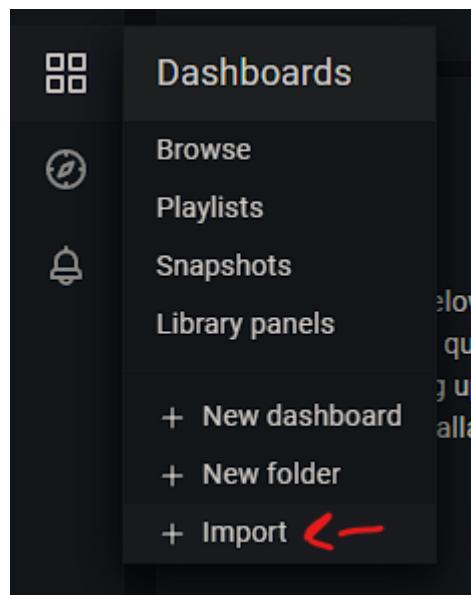
Observabilidad con Grafana

Codigos de las plantillas para Grafana:

Elasticsearch: 14191 MariaDB: 13106 Rabbit: 10991

Aplicar plantillas con codigos

Para aplicar las plantillas anteriores, hay que primero, ingresar a grafana, ir al icono de cuatro cuadrados y presionar import como en la siguiente imagen:



Una vez en ese lugar, colocar uno de los anteriores y presionar load:

Journarl Search Platform

The screenshot shows the Journarl Search Platform interface. At the top, there are four navigation items: "Browse", "Playlists", "Snapshots", and "Library panels". Below them is a blue button labeled "Upload JSON file" with an upward arrow icon. Underneath this is a section titled "Import via grafana.com" containing a text input field with the value "14191" and a blue "Load" button to its right, which is circled in red. Below this section is another titled "Import via panel json" with a blue "Load" button at the bottom.

Por ultimo presionar el boton que dice import:

Importing dashboard from [Grafana.com](#)

Published by	Grafana Labs
Updated on	2021-04-06 10:56:02

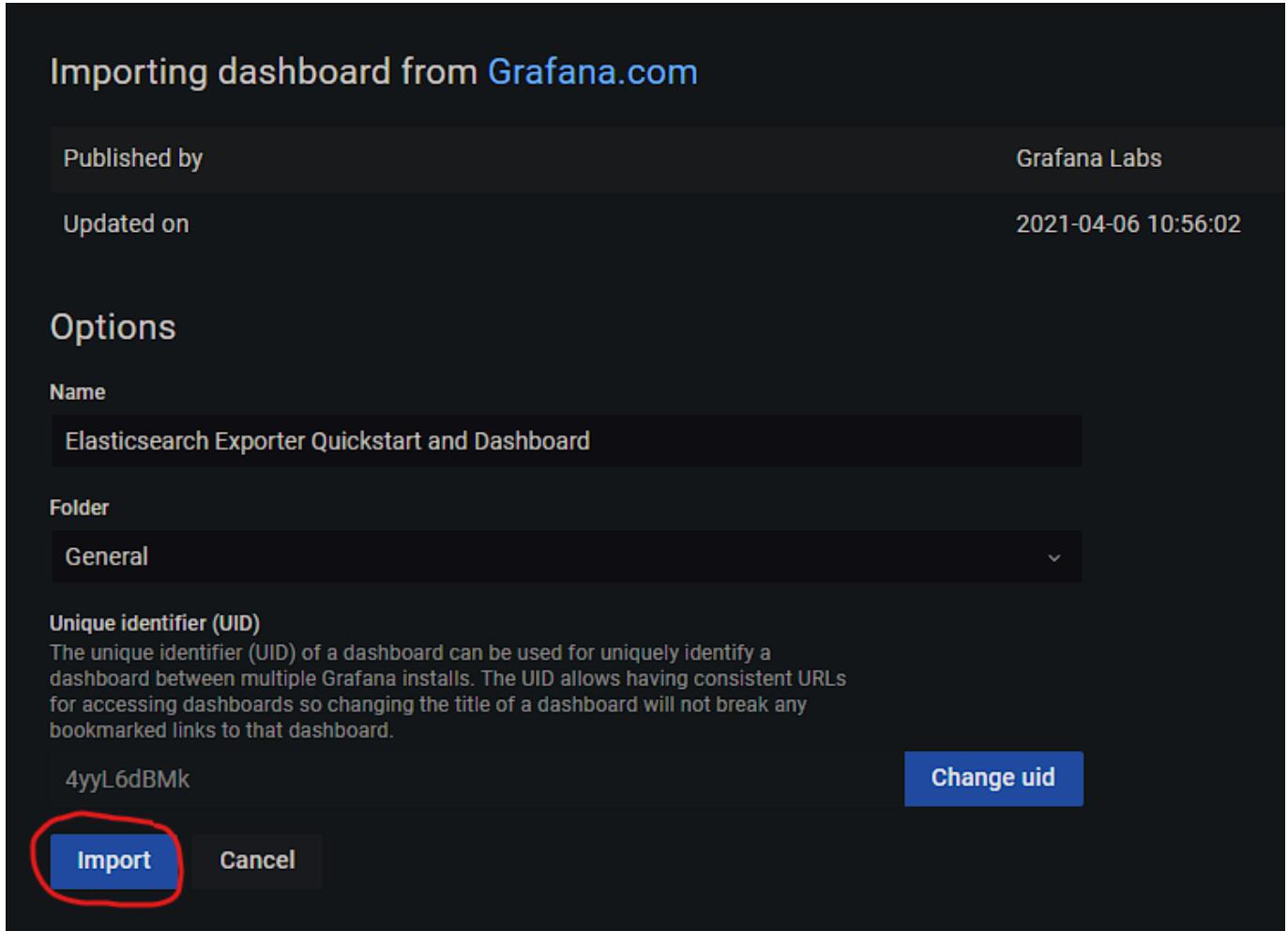
Options

Name: Elasticsearch Exporter Quickstart and Dashboard

Folder: General

Unique identifier (UID): 4yyL6dBMc
[Change uid](#)

Import (button circled in red) [Cancel](#)

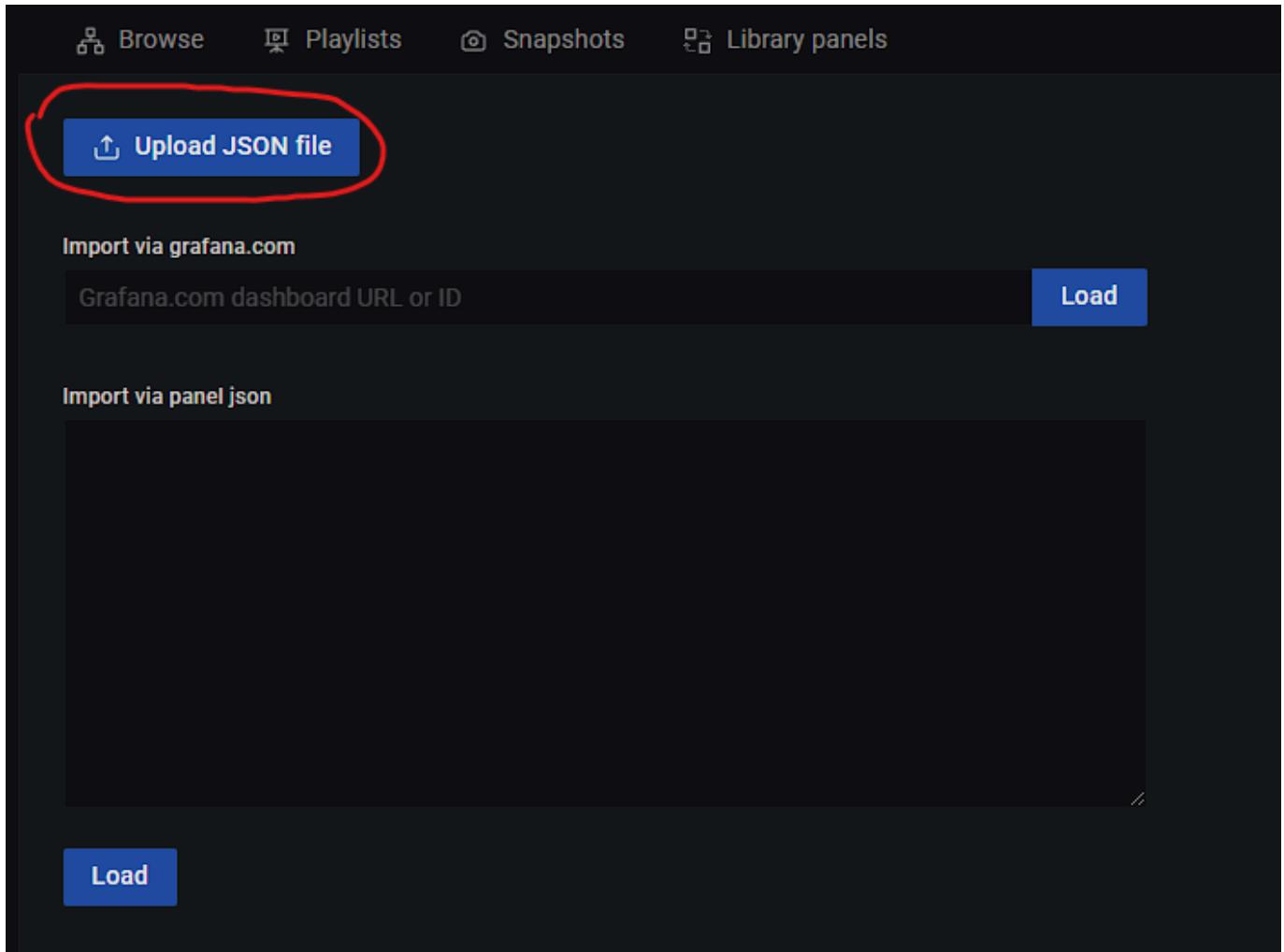


Repetir los pasos con cada código.

Importar plantilla de la App desarrollada

La plantilla de la app desarrollada se importa de una forma parecida, después del segundo paso hay que hacer lo siguiente, pulsar en el botón que dice "Upload JSON file":

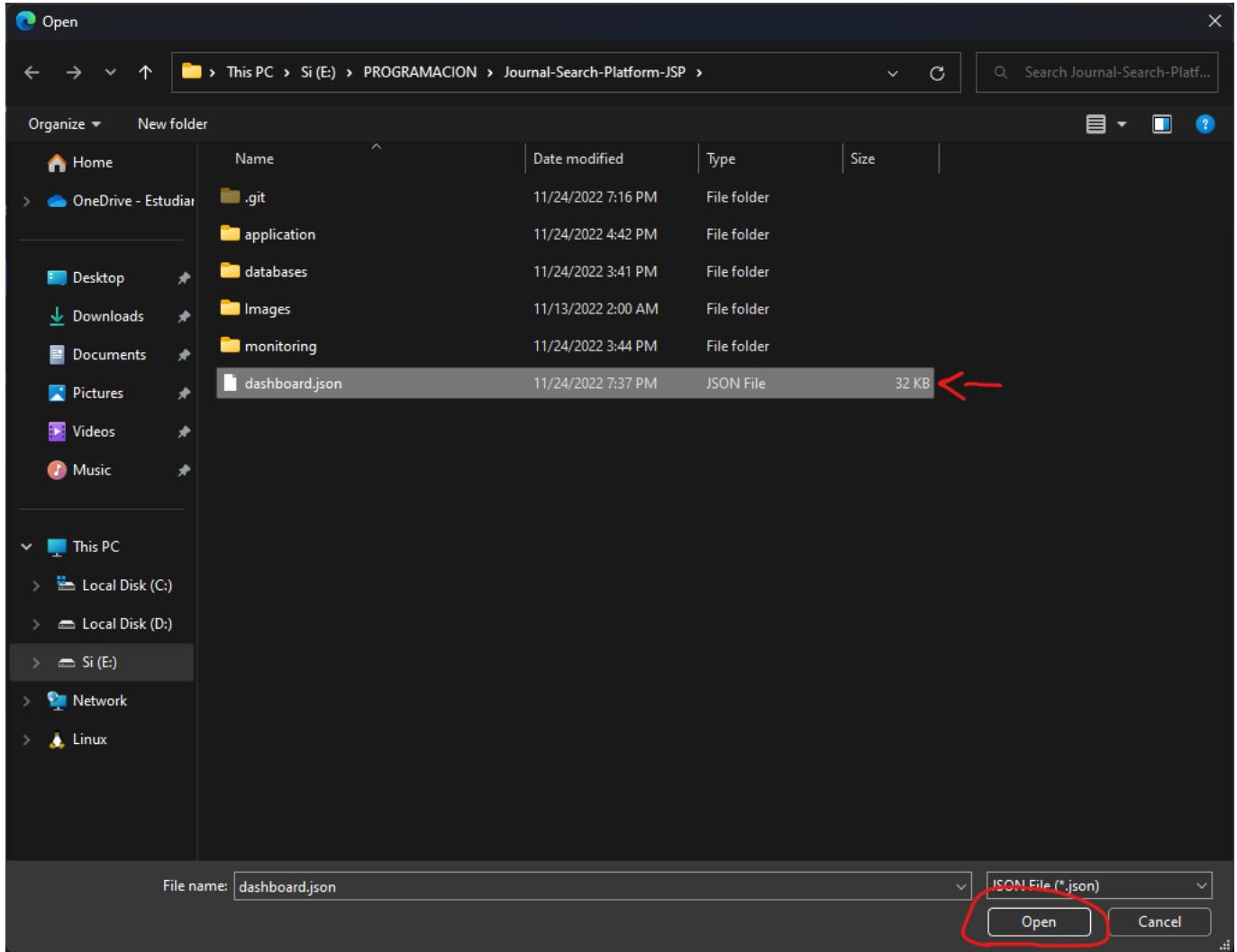
Journarl Search Platform



The screenshot shows the Journarl Search Platform interface. At the top, there is a navigation bar with four items: "Browse", "Playlists", "Snapshots", and "Library panels". Below the navigation bar, there is a blue button with a white arrow icon and the text "Upload JSON file". This button is circled in red. To the right of this button is a section titled "Import via grafana.com" with a text input field for "Grafana.com dashboard URL or ID" and a blue "Load" button. Below this section is another titled "Import via panel json" with a large, empty dark rectangular area for pasting JSON. At the bottom left of this area is a blue "Load" button.

Una vez hecho eso, seleccionar el json llamado "dashboard.json" ubicado en la raiz de este proyecto y abrirlo:

Journarl Search Platform



Por ultimo, presionar importar:

The screenshot shows the 'Options' dialog box from the Journarl Search Platform. At the top, there are navigation links: 'Browse', 'Playlists', 'Snapshots', and 'Library panels'. The main area is titled 'Options' and contains the following fields:

- Name:** JSP-Application
- Folder:** General
- Unique identifier (UID):** 80diyUOVka

At the bottom right are two buttons: 'Import' (highlighted with a red circle) and 'Cancel'.

Values de Application

El archivo values.yaml que se puede encontrar en la carpeta ./application contiene la información que necesitan componentes como ElasticSearch, RabbitMQ, MariaDB. Para estos se tienen valores **generales** que debe utilizar cada aplicación del sistema, estas son:

```
Host: Dirección del proveedor.  
Port: Puerto de conexión.  
User: Usuario con el cual brinda el acceso al servicio.
```

Lo que podemos encontrar dentro del archivo es:

```

general:
  # Elastic
  elastichost: "http://databases-elasticsearch-master-h1.default.svc.cluster.local"
  elasticport: 9200
  elasticuser: "elastic"
  elasticindex: "registries"

  # RabbitMQ
  rabbithost: "databases-rabbitmq-headless"
  rabbitport: 5672
  rabbituser: "user"

  # Mariadb
  mariadbname: "my_database"
  mariadbhost: "databases-mariadb-primary"
  mariadbport: 3306
  mariadbuser: "root"
  apiurl: "https://api.biorxiv.org/"

```

Ahora, con los componentes creados por el equipo se tienen valores para cada uno de ellos, estos consisten en: cantidad de réplicas y nombre de la cola de RabbitMQ en caso de ser necesario. Un ejemplo práctico sería el siguiente caso:

```

loader:
  queuename: "loader"
  sleeptime: 1
  replicas: 3
  metrics: 6943

```

Al final este archivo nos ayuda en la configuración de cada uno de los componentes y especificar la cantidad de réplicas que se desean de cierto componente.

NodePorts

Los servicios de tipo NodePort nos permiten comunicar componentes dentro del cluster de Kubernetes con los componentes fuera de este ambiente, esto es muy funcional para poder visualizar distintos puntos desde aplicaciones terceras o tener la seguridad de un host y port donde vamos a poder consultar por datos. Para poder crear un servicio de tipo NodePort se tienen dos opciones, crear un servicio mediante un archivo yaml o mediante la configuración en el values dentro del Helm chart correspondiente. En nuestro grupo de trabajo se escogió la segunda opción.

Esto se ve de la siguiente manera dentro del values del Helm Chart databases:

```

kibana:
  service:
    type: NodePort
    nodePorts:
      http: 31500

```

Este es un ejemplo de los varios que hay dentro del documento, en este caso, Kibana. Para poder crear este servicio se necesita:

- Especificar el servicio.
- Especificar el tipo de servicio, en este caso NodePort.
- Especificar el puerto a exponer, en este caso fue el 31500. Los puertos deben estar dentro de un rango de [30000,32767].

Dentro del proyecto todos los componentes de monitoring y databases tienen sus propios NodePorts para un mejor manejo durante la etapa de desarrollo, sin embargo, si el usuario desea seguir con el clásico ClusterIP se debe cambiar simplemente el tipo de servicio.

```
type: NodePort --> type: ClusterIP
```

Recursos

Limitación de recursos, la elaboración del proyecto fue un poco difícil ya que todos los integrantes no poseen la facilidad de un equipo con los recursos suficientes para poder levantar todos los componentes que se necesitaban dentro del proyecto, es por eso que se optó por ciertas opciones que ayudaban a reducir el consumo de memoria, una de las soluciones para este tipo de situaciones es la limitación de los componentes, en términos generales esto consiste en darle valores máximos de uso de los recursos a cada una de las aplicaciones que se encuentran corriendo, lo que nos ayuda a la reducción general del uso de memoria.

Para cada componente que se quiera limitar se sigue una plantilla general:

```
resources:
  limits:
    cpu: 100m
    memory: 128Mi
  requests:
    cpu: 100m
    memory: 128Mi
```

Donde los valores ingresados son medidos por:

- m --> milésima de núcleo (thousandth of a core)
- Mi --> Mibibyte

De esta manera se puede ir por cada componente hasta llegar un balance del consumo de recursos en de uso y de request para un mejor rendimiento.

Contraseñas

Los componentes de ElasticSearch, MariaDB y RabbitMQ poseen sus contraseñas, en nuestro equipo de trabajo no se vio necesaria la estipulación de contraseñas definidas, sino que, por el contrario, cada uno de los componentes genera su propio usuario y contraseña. A raíz de esta situación es que antes de realizar alguna de las pruebas que se documentan en esta sección debe tener presente esto. Al momento de instalar los componentes debe recuperar las contraseñas e insertarlas en los espacios correspondientes, para esto se deben seguir los siguientes pasos:

- Utilizar la aplicación Lens para una mejor comodidad.
- Navegar a la conexión del cluster, en la sección "Config" se encuentra un espacio de nombre "Secrets".

Journarl Search Platform

Name	Namespace	Labels	Keys	Type	Age
alertmanager-main-alertmanager	default	app.kubernetes.io/component=alertmanager	alertmanager.yaml	Opaque	3h13m
alertmanager-main-alertmanager-generated	default	managed-by=prometheus-operator	alertmanager.yaml.gz	Opaque	3h13m
alertmanager-main-alertmanager-tls-assets-0	default	managed-by=prometheus-operator		Opaque	3h13m
alertmanager-main-alertmanager-web-config	default	managed-by=prometheus-operator	web-config.yaml	Opaque	3h13m
databases-elasticsearch	default	app.kubernetes.io/instance=datab	elasticsearch-password	Opaque	3h13m
databases-elasticsearch-coordinating crt	default	app.kubernetes.io/component=crt	ca.crt, tls.crt, tls.key	kubernetes.io/tls	3h13m
databases-elasticsearch-data crt	default	app.kubernetes.io/component=crt	ca.crt, tls.crt, tls.key	kubernetes.io/tls	3h13m
databases-elasticsearch-master crt	default	app.kubernetes.io/component=crt	ca.crt, tls.crt, tls.key	kubernetes.io/tls	3h13m
databases-kibana	default	app.kubernetes.io/instance=datab	kibana-password	Opaque	3h13m
databases-mariadb	default	app.kubernetes.io/instance=datab	mariadb-replication-password, _	Opaque	3h13m
monitoring-grafana-admin	default	app.kubernetes.io/component=gra	GF_SECURITY_ADMIN_PASSWORD	Opaque	3h13m
monitoring-grafana-datasources	default	app.kubernetes.io/component=gra	datasources.yaml	Opaque	3h13m
monitoring-rabbitmq	default	app.kubernetes.io/instance=monit	rabbitmq-erlang-cookie, rabbitm...	Opaque	3h13m
monitoring-rabbitmq-config	default	app.kubernetes.io/instance=monit	rabbitmq.conf	Opaque	3h13m
prometheus-main-prometheus	default	managed-by=prometheus-operator	prometheus.yaml.gz	Opaque	3h13m
prometheus-main-prometheus-tls-assets-0	default	managed-by=prometheus-operator		Opaque	3h13m
prometheus-main-prometheus-web-config	default	managed-by=prometheus-operator	web-config.yaml	Opaque	3h13m
quickstart-kibana-user	default	eck.k8s.elastic.co/credentials-tru	hash, name, serviceAccount, tok...	Opaque	55d
sh.helm.release.v1.application.v1	default	modifiedAt=166043532	name: release	helm.sh/release.v1	52m
sh.helm.release.v1.databases.v1	default	modifiedAt=166035084	name: release	helm.sh/release.v1	3h13m
sh.helm.release.v1.monitoring.v1	default	modifiedAt=166035093	name: release	helm.sh/release.v1	3h13m

- Una vez en esta sección se pueden ver todos los secrets de todos los componentes, así que de esta manera podemos seleccionar el componente en cuestión, obtener sus información de acceso y sustituirla en los archivos de prueba que se mencionan en las secciones siguientes.

Para continuar con el ejemplo, se obtendrá la contraseña de ElasticSearch, esta se puede encontrar seleccionando el componente en la sección de secrets vista, por lo que se obtiene lo siguiente:

Secret: databases-elasticsearch

Created	3h 23m ago 2022-10-17T13:31:24-06:00
Name	databases-elasticsearch
Namespace	default
Labels	app.kubernetes.io/instance=databases app.kubernetes.io/managed-by=Helm app.kubernetes.io/name=elasticsearch helm.sh/chart=elasticsearch-19.3.0
Annotations	meta.helm.sh/release-name=databases meta.helm.sh/release-namespace=default
Type	Opaque
Data	
elasticsearch-password	VtIRxwYQdxIPJz93

Save

La contraseña encontrada se escribirá dentro del espacio correspondiente en el archivo de pruebas como se ve a continuación:

```

7 # Elastic
8 ELASTICHOST = "localhost"
9 ELASTICPORT = "32500"
10 ELASTICUSER = "elastic"
11 ELASTICPASS = "VtIRxwYQdxIPJz93"

```

Note la línea azul de la izquierda, en ese punto es dónde se ha inscrito la contraseña.

Debe tener esto en cuenta antes de replicar cada una de las pruebas que en este documento se detallan.

NOTA

Al momento de realizar la instalación de cada uno de los componentes crea un *Persistent Volume Claim* (PVC) y un *Persistent Volume* (PV) dónde se guarda la información que contienen en ese momento de manera persistente, al momento de desinstalar

Journarl Search Platform

los componentes estas secciones no se eliminan, es por eso que si al reinstalar los componentes y realizar los pasos anteriores no funcionan, asegúrese de haber eliminado previamente los PVC y PV antes de volver a instalar los componentes, pues la información de la contraseña que se está mostrando no es la correcta, ya que se muestra la contraseña actual, pero los componentes están tomando como contraseña la que dice el PVC y el PV, que son de instalaciones previas. Con la ayuda de Lens es muy fácil realizar este proceso con los siguientes pasos:

- En la sección de Storage, seleccionar el espacio de PVC.
- Seleccionar los elementos a eliminar.
- Eliminarlos.

The screenshot shows the Journarl Search Platform interface. On the left, there is a sidebar with various navigation options. A red box highlights the 'Persistent Volume Claims' option under the 'Storage' section, which is labeled with a '1'. In the center, a table titled 'Persistent Volume Claims' displays eight items. The table has columns for Name, Namespace, Storage class, Size, Pods, Age, and Status. Each row shows a different PVC name, its namespace (default), storage class (hostpath), size (8Gi or 10Gi), associated pods, age (3h38m), and status (Bound). A large number '2' is placed above the table. A small circular icon with a minus sign is located at the bottom right of the table area, labeled with a '3'.

Name	Namespace	Storage class	Size	Pods	Age	Status
data-databases-elasticsearch-data-0	default	hostpath	8Gi	databases-elasticsearch-data-0	3h38m	Bound
data-databases-elasticsearch-data-1	default	hostpath	8Gi	databases-elasticsearch-data-1	3h38m	Bound
data-databases-elasticsearch-master-0	default	hostpath	8Gi	databases-elasticsearch-master-0	3h38m	Bound
data-databases-mariadb-primary-0	default	hostpath	8Gi	databases-mariadb-primary-0	3h38m	Bound
data-databases-mariadb-secondary-0	default	hostpath	8Gi	databases-mariadb-secondary-0	3h38m	Bound
data-monitoring-rabbitmq-0	default	hostpath	8Gi	monitoring-rabbitmq-0	3h38m	Bound
databases-kibana	default	hostpath	10Gi	databases-kibana-5f598c65-qctcz	3h38m	Bound
monitoring-grafana	default	hostpath	10Gi	monitoring-grafana-56df5b79bf-vzfx2	3h38m	Bound

Pruebas realizadas

Pruebas Generales

Para la ejecución de las pruebas generales se necesita tener todos los componentes previamente instalados. Con el comando

```
kubectl get pods
```

Si los pasos de instalación fueron ejecutados correctamente se debe obtener el siguiente resultado:

Journarl Search Platform

NAME	READY	STATUS	RESTARTS	AGE
alertmanager-main-alertmanager-0	2/2	Running	0	165m
api-9f687c7-vfbmf	1/1	Running	0	88s
databases-elasticsearch-coordinating-0	1/1	Running	3 (8h ago)	2d3h
databases-elasticsearch-data-0	1/1	Running	2 (8h ago)	2d3h
databases-elasticsearch-data-1	1/1	Running	2 (8h ago)	2d3h
databases-elasticsearch-master-0	1/1	Running	2 (8h ago)	2d3h
databases-elasticsearch-metrics-796d5cf8-8k9kc	1/1	Running	2 (8h ago)	2d3h
databases-kibana-7f75698cf6-qnhwz	1/1	Running	3 (8h ago)	2d3h
databases-mariadb-primary-0	2/2	Running	7 (8h ago)	2d3h
databases-mariadb-secondary-0	2/2	Running	4 (8h ago)	2d3h
databases-rabbitmq-0	1/1	Running	2 (8h ago)	2d3h
details-downloader-56748d974d-9tk9h	1/1	Running	0	88s
details-downloader-56748d974d-bq275	1/1	Running	0	88s
details-downloader-56748d974d-jbmrxr	1/1	Running	0	88s
downloader-68949f98c9-bm8jn	1/1	Running	0	88s
downloader-68949f98c9-f6597	1/1	Running	0	88s
downloader-68949f98c9-gm29r	1/1	Running	0	88s
jatsxml-processor-c594ff4f5-65gj6	1/1	Running	0	88s
jatsxml-processor-c594ff4f5-j75h5	1/1	Running	0	88s
jatsxml-processor-c594ff4f5-kmf2	1/1	Running	0	88s
loader-processor-6876495585-jchpq	1/1	Running	0	88s
loader-processor-6876495585-tpvjq	1/1	Running	0	88s
loader-processor-6876495585-xzdmd	1/1	Running	0	88s
main-blackbox-exporter-7fb5d446fd-klxj7	1/1	Running	0	165m
main-operator-7969c544fd-tt2tz	1/1	Running	0	165m
monitoring-grafana-5d44bd48cb-v8bdj	1/1	Running	0	101m
monitoring-kube-state-metrics-56c6854f9b-jhm9r	1/1	Running	0	165m
monitoring-node-exporter-xrqzx	1/1	Running	0	165m
prometheus-main-prometheus-0	2/2	Running	0	165m

Esto indica que se tienen todos los componentes del proyecto instalados de buena manera y funcionando.

Despues hay que realizar los pasos para la ejecucion del proyecto que se encuentra en la seccion [Ejecución del proyecto](#)

Una vez con todos los componentes instalados de buena manera y se realizaron los pasos para la ejecucion del proyecto, se procede con la prueba general de funcionamiento:

Prueba del Pipeline:

Para probar el pipeline se necesita crear un job, esto se hace de la siguiente manera. En el app se inicia sesion (se registra previamente):



JSP

Journal Search Platform

deanyt0417@gmail.com

.....

Iniciar Sesión

Registrarse

Luego hay que seleccionar la opcion "Crear job":

Opciones

Crear job



Buscar artículo

Ver artículos
guardados

Por ultimo, colocar un valor para el tamaño de los grupos y presionar en crear:

Crear job

50 ←

Crear



<< Volver

Resultados:

Esto procede a crear un job en mariaDB de la siguiente manera:

Journarl Search Platform

	id	created	status	end	loader	grp_size
▶	8	2022-11-24 21:31:17	In-progress	0000-00-00 00:00:00	loader-processor-6876495585-tpvjjz	50
*		HULL	HULL	HULL	HULL	HULL

El loader crea los grupos y los demás componentes empiezan a trabajar sobre ellos:

	id	id_job	created	end	stage	grp_number	status	offset
▶	7828	8	2022-11-24 21:31:18	HULL	details-downloader	0	in-progress	0
	7829	8	2022-11-24 21:31:18	HULL	downloader	1	completed	50
	7830	8	2022-11-24 21:31:18	HULL	details-downloader	2	in-progress	100
	7831	8	2022-11-24 21:31:18	HULL	downloader	3	completed	150
	7832	8	2022-11-24 21:31:18	HULL	details-downloader	4	in-progress	200
	7833	8	2022-11-24 21:31:18	HULL	downloader	5	completed	250
	7834	8	2022-11-24 21:31:18	HULL	downloader	6	completed	300
	7835	8	2022-11-24 21:31:18	HULL	downloader	7	completed	350
	7836	8	2022-11-24 21:31:18	HULL	downloader	8	completed	400
	7837	8	2022-11-24 21:31:18	HULL	downloader	9	completed	450
	7838	8	2022-11-24 21:31:18	HULL	downloader	10	completed	500
	7839	8	2022-11-24 21:31:18	HULL	downloader	11	completed	550
	7840	8	2022-11-24 21:31:18	HULL	downloader	12	completed	600
	7841	8	2022-11-24 21:31:18	HULL	downloader	13	completed	650
	7842	8	2022-11-24 21:31:18	HULL	downloader	14	completed	700
	7843	8	2022-11-24 21:31:18	HULL	downloader	15	completed	750
	7844	8	2022-11-24 21:31:18	HULL	downloader	16	in-progress	800
	7845	8	2022-11-24 21:31:18	HULL	downloader	17	in-progress	850
	7846	8	2022-11-24 21:31:18	HULL	downloader	18	in-progress	900
	7847	8	2022-11-24 21:31:18	HULL	Loader	19	HULL	950
	7848	8	2022-11-24 21:31:18	HULL	Loader	20	HULL	1000
	7849	8	2022-11-24 21:31:18	HULL	Loader	21	HULL	1050
	7850	8	2022-11-24 21:31:18	HULL	Loader	22	HULL	1100
	7851	8	2022-11-24 21:31:18	HULL	Loader	23	HULL	1150
	7852	8	2022-11-24 21:31:18	HULL	Loader	24	HULL	1200
	7853	8	2022-11-24 21:31:18	HULL	Loader	25	HULL	1250
	7854	8	2022-11-24 21:31:18	HULL	Loader	26	HULL	1300

	id	id_job	created	end	stage	grp_number	status	offset
▶	7828	8	2022-11-24 21:31:18	2022-11-24 21:33:40	jatsxml-processor	0	Completed	0
	7829	8	2022-11-24 21:31:18	2022-11-24 21:35:14	jatsxml-processor	1	Completed	50
	7830	8	2022-11-24 21:31:18	2022-11-24 21:33:42	jatsxml-processor	2	Completed	100
	7831	8	2022-11-24 21:31:18	2022-11-24 21:35:21	jatsxml-processor	3	Completed	150
	7832	8	2022-11-24 21:31:18	2022-11-24 21:33:34	jatsxml-processor	4	Completed	200
	7833	8	2022-11-24 21:31:18	2022-11-24 21:36:52	jatsxml-processor	5	Completed	250
	7834	8	2022-11-24 21:31:18	2022-11-24 21:37:01	jatsxml-processor	6	Completed	300
	7835	8	2022-11-24 21:31:18	2022-11-24 21:35:23	jatsxml-processor	7	Completed	350
	7836	8	2022-11-24 21:31:18	2022-11-24 21:37:04	jatsxml-processor	8	Completed	400
	7837	8	2022-11-24 21:31:18	HULL	jatsxml-processor	9	In-progress	450
	7838	8	2022-11-24 21:31:18	HULL	jatsxml-processor	10	In-progress	500
	7839	8	2022-11-24 21:31:18	HULL	jatsxml-processor	11	In-progress	550
	7840	8	2022-11-24 21:31:18	HULL	details-downloader	12	Completed	600
	7841	8	2022-11-24 21:31:18	HULL	details-downloader	13	Completed	650
	7842	8	2022-11-24 21:31:18	HULL	details-downloader	14	Completed	700
	7843	8	2022-11-24 21:31:18	HULL	details-downloader	15	Completed	750
	7844	8	2022-11-24 21:31:18	HULL	details-downloader	16	Completed	800
	7845	8	2022-11-24 21:31:18	HULL	details-downloader	17	Completed	850
	7846	8	2022-11-24 21:31:18	HULL	details-downloader	18	Completed	900

Journal Search Platform

En la tabla history se empieza a reflejar los procesos que realizan los pods:

	id	component	status	created	end	message	grp_id	stage
▶	5389	downloader-68949f98c9-bm8jn	completed	2022-11-24 21:31:18	2022-11-24 21:31:26	successful process	7829	downloader
	5390	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:18	2022-11-24 21:31:23	successful process	7830	downloader
	5391	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:18	2022-11-24 21:31:23	successful process	7828	downloader
	5392	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:23	2022-11-24 21:31:26	successful process	7831	downloader
	5393	details-downloader-56748d974d-bq275	Completed	2022-11-24 21:31:23	2022-11-24 21:32:00	successful process	7828	details-downloader
	5394	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:23	2022-11-24 21:31:26	successful process	7832	downloader
	5395	details-downloader-56748d974d-9tk9h	Completed	2022-11-24 21:31:23	2022-11-24 21:32:00	successful process	7830	details-downloader
	5396	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:26	2022-11-24 21:31:31	successful process	7833	downloader
	5397	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:26	2022-11-24 21:31:32	successful process	7834	downloader
	5398	details-downloader-56748d974d-jbmxr	Completed	2022-11-24 21:31:26	2022-11-24 21:32:02	successful process	7832	details-downloader
	5399	downloader-68949f98c9-bm8jn	completed	2022-11-24 21:31:26	2022-11-24 21:31:30	successful process	7835	downloader
	5400	downloader-68949f98c9-bm8jn	completed	2022-11-24 21:31:30	2022-11-24 21:31:33	successful process	7836	downloader
	5401	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:31	2022-11-24 21:31:34	successful process	7837	downloader
	5402	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:32	2022-11-24 21:31:35	successful process	7838	downloader
	5403	downloader-68949f98c9-bm8jn	completed	2022-11-24 21:31:33	2022-11-24 21:31:36	successful process	7839	downloader
	5404	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:34	2022-11-24 21:31:37	successful process	7840	downloader
	5405	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:35	2022-11-24 21:31:38	successful process	7841	downloader
	5406	downloader-68949f98c9-bm8jn	completed	2022-11-24 21:31:36	2022-11-24 21:31:40	successful process	7842	downloader
	5407	downloader-68949f98c9-f6597	completed	2022-11-24 21:31:37	2022-11-24 21:31:41	successful process	7843	downloader
	5408	downloader-68949f98c9-gm29r	completed	2022-11-24 21:31:38	2022-11-24 21:31:42	successful process	7844	downloader

Se crea un indice donde se guardan los documentos con el nombre especificado en el values.yaml:

```
10 "hits": {
11   "total": {
12     "value": 726,
13     "relation": "eq"
14   },
15   "max_score": 1,
16   "hits": [
17     {
18       "_index": "registries",
19       "_id": "7QXXrIQBus4QMSxCzO2u",
20       "_score": 1,
21       "_ignored": [
22         "rel_abs.keyword",
23         "details.authors.keyword",
24         "details.abstract.keyword"
25       ],
26       "_source": {
27         "rel_doi": "10.1101/2022.11.22.517465",
28         "rel_title": "Nirmatrelvir treatment blunts the development of antiviral adaptive immune responses
29           in SARS-CoV-2 infected mice",
30         "rel_date": "2022-11-22",
31         "rel_site": "bioRxiv",
32         "rel_link": "https://www.biorxiv.org/cgi/content/short/2022.11.22.517465",
33         "rel_abs": "Alongside vaccines, antiviral drugs are becoming an integral part of our response to
34           the SARS-CoV-2 pandemic. Nirmatrelvir - an orally available inhibitor of the 3-chymotrypsin-like
35           cysteine protease - has been shown to reduce the risk of progression to severe COVID-19. However,
36           the impact of nirmatrelvir treatment on the development of SARS-CoV-2-specific adaptive immune
37           responses is unknown. Here, by using a mouse model of SARS-CoV-2 infection, we show that
38           nirmatrelvir administration early after infection blunts the development of SARS-CoV-2-specific
39           antibody and T cell responses. Accordingly, upon secondary challenge, nirmatrelvir-treated mice
40           recruited significantly fewer memory T and B cells to the infected lungs and to mediastinal lymph
41           nodes, respectively. Together, the data highlight a potential negative impact of nirmatrelvir
42           treatment with respect to adaptive immune responses to SARS-CoV-2 infection."}
43     }
44   ]
45 }
```

En el indice de groups se puede ver como van bajando la cantidad de groups en el indice:

Journal Search Platform

```
10 "hits": {
11   "total": {
12     "value": 497,
13     "relation": "eq"
14   },
15   "max_score": 1,
16   "hits": [
17     {
18       "_index": "groups",
19       "_id": "OAXYrIQBus4QMSxCP-6s",
20       "_score": 1,
21       "_ignored": [
22         "docs.rel_abs.keyword"
23       ],
24       "_source": {
25         "id_job": "8",
26         "grp_number": "65",
27         "docs": [
28           {
29             "rel_doi": "10.1101/2022.03.07.481785",
30             "rel_title": "SARS-CoV-2 Spike evolution influences GBP and IFITM sensitivity",
31             "rel_date": "2022-03-07",
32             "rel_site": "bioRxiv",
33             "rel_link": "https://www.biorxiv.org/cgi/content/short/2022.03.07.481785",
34             "rel_abs": "SARS-CoV-2 spike requires proteolytic processing for viral entry. The presence of a polybasic furin-cleavage site (FCS) in spike, and evolution towards an optimised FCS by dominant variants of concern (VOCs), are linked to enhanced infectivity and transmission. Here we show that interferon-inducible antiviral restriction factors Guanylate binding proteins (GBP) 2 and 5 interfere with furin-mediated cleavage of SARS-CoV-2 spike and inhibit the infectivity of early-lineage Wuhan-Hu-1, while VOCs Alpha and Delta have evolved to escape restriction. Strikingly, we find Omicron is unique amongst VOCs, being restricted by GBP2/5 and the IFITM1, Goblet, Poliovirus-like 55 kDa protein, all three of which are still present in the Omicron spike protein."}
35         ]
36       }
37     }
38   ]
39 }
```

En las colas de RabbitMQ se puede ver como trabajan:

Overview				Messages			Message rates			+/-
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack	
details-downloader	classic		idle	39	3	42	0.20/s	0.00/s	0.00/s	
downloader	classic		running	416	3	419	0.80/s	0.20/s	0.20/s	
loader	classic		running	34	3	37	0.00/s	0.80/s	0.80/s	

Pruebas de búsquedas en la App:

Para probar las búsquedas en la app ahora se debe elegir la opción de "Buscar artículo":

Opciones

Crear job

Buscar artículo

Ver artículos
guardados



Sera presentado con la siguiente pantalla, solo queda colocar algo para buscar, ya sea nombre de autores, fechas, titulos, etc. y presionar "Buscar":



Search

Buscar

<< Volver

Resultados

Buscar "covid" da los siguientes resultados:

1 de 10

What is the effectiveness of financial support schemes for individuals requested to self-isolate following a positive Covid test or positive contact: A rapid review >

Long Covid active case finding: a co-produced community-based pilot within the STIMULATE-ICP study (Symptoms, Trajectory, Inequalities and Management: Understanding Long-COVID to Address and Transform Existing Integrated Care Pathways) >

A rapid review of Supplementary air filtration systems in health service settings. September 2022. >

Conceptualising the Episodic Nature of Disability among Adults Living with Long COVID: A Qualitative Study >

The Impact of State Paid Sick Leave Policies on Longitudinal Weekday Workplace Mobility During the COVID-19 Pandemic >

Characterising Long Covid: a living systematic review update with controlled studies >

<< Volver

Siguiente

Buscar "2011" da los siguientes resultados:

1 de 1

Potential risk polarization for acute myocardial infarction
during the COVID-19 pandemic: Single-center experiences in
Osaka, Japan >

What is the effectiveness of financial support schemes for
individuals requested to self-isolate following a positive
Covid test or positive contact: A rapid review >

<< Volver

Siguiente

Buscar "issue" da los siguientes resultados:

1 de 2

Differential Impacts of Perceived Social Support on Alcohol and Cannabis Use in Young Adults: Lessons from the COVID-19 Pandemic >

Use of the particle agglutination/particle agglutination-inhibition test for antigenic analysis of SARS-CoV-2 >

Brief report: Preterm birth rates among twins during the Danish COVID-19 lockdown and mitigation period. >

FULL-SCALE DEEPLY SUPERVISED ATTENTION NETWORK FOR SEGMENTING COVID-19 LESIONS >

Antimicrobial copper as an effective and practical deterrent to surface transmission of SARS-CoV-2 >

Covid-19 pandemic has polarized the society toward negative and positive traits depending on a person's resilience and certain predispositions >

<< Volver

Siguiente

Prueba artículos guardados:

Estando en la lista de resultados de la prueba anterior, se puede presionar un artículo y se le presenta lo siguiente:

Artículo seleccionado

Título

Potential risk polarization for acute myocardial infarction during the COVID-19 pandemic: Single-center experiences in Osaka, Japan

Abstract

This study compared the time course and outcomes of acute myocardial infarction, including mechanical complications and hospital mortality, before and after the coronavirus disease 2019 (COVID-19) pandemic at a regional core hospital in South Osaka, Japan. Moreover, it identified predictors for hospital mortality and mechanical complications. In total, 503 patients who underwent emergency percutaneous coronary intervention between January 2011 and December 2020.

Autor(es)

Masato Furui >

Kenji Kawajiri >

<< Volver

Like

Al dar like pasa lo siguiente:

Artículo seleccionado

Título

Potential risk polarization for acute myocardial infarction during the COVID-19 pandemic: Single-center experiences in Osaka, Japan

Abstract

This study compared the time course and outcomes of acute myocardial infarction, including mechanical complications and hospital mortality, before and after the coronavirus disease 2019 (COVID-19) pandemic at a regional core hospital in South Osaka, Japan. Moreover, it identified predictors for hospital mortality and mechanical complications. In total, 503 patients who underwent emergency percutaneous coronary intervention between January 2011 and December 2021 at our institution were examined retrospectively.

Autor(es)

Masato Furui >

Kenji Kawajiri >

Guardado correctamente

<< Volver

Me gusta

Ahora eligiendo la opcion "Ver artículos guardados" en el menu de opciones podra ver los articulos que guardo:

Opciones

Crear job

Buscar artículo

Ver artículos
guardados



Resultados:

Se tienen los siguientes artículos guardados:

Artículos que me gustaron

Excess death estimates from multiverse analysis
in 2009-2021 >

Potential risk polarization for acute myocardial
infarction during the COVID-19 pandemic:
Single-center experiences in Osaka, Japan >

<< Volver

Presionar alguno, despliega lo siguiente:

Detalles del artículo

Titulo

Potential risk polarization for acute myocardial infarction during the COVID-19 pandemic: Single-center experiences in Osaka, Japan

Abstract

This study compared the time course and outcomes of acute myocardial infarction, including mechanical complications and hospital mortality, before and after the coronavirus disease 2019 (COVID-19) pandemic at a regional core hospital in South Osaka, Japan. Moreover, it identified predictors for hospital mortality and mechanical complications. In total, 503 patients who underwent emergency percutaneous coronary intervention between January 2011 and December 2021 at our institution were examined retrospectively. The time course of acute

Autor(es)

Masato Furui >

Kenji Kawajiri >

Takeshi Yoshida >

<< Volver

Pruebas unitarias y resultados de estas.

Pruebas Unitarias de Loader

Journarl Search Platform

Este componente es el encargado de crear la cantidad de grupos especificada por un tamaño de grupo. Este componente recibe un conjunto de datos conocido como *Job* el cual contiene los datos necesarios para el procesamiento del mismo.

Los Jobs se guardan en una base de datos destinada para ellos la cual es MariaDB y se guardan de la siguiente manera:

id	created	status	end	loader	grp_size
1	2022-11-23 19:49:13	NEW	NULL	NULL	1684
2	2022-11-23 19:49:13	NEW	NULL	NULL	808
3	2022-11-23 19:49:13	NEW	NULL	NULL	1233
4	2022-11-23 19:49:13	NEW	NULL	NULL	1563
5	2022-11-23 19:49:13	NEW	NULL	NULL	1951
6	2022-11-23 19:49:13	NEW	NULL	NULL	1786
7	2022-11-23 19:49:13	NEW	NULL	NULL	1681
8	2022-11-23 19:49:13	NEW	NULL	NULL	925
9	2022-11-23 19:49:13	NEW	NULL	NULL	166
10	2022-11-23 19:49:13	NEW	NULL	NULL	1722
11	2022-11-23 19:49:13	NEW	NULL	NULL	314
12	2022-11-23 19:49:13	NEW	NULL	NULL	963

El trabajo real del Loader se consta en:

- tomar uno a uno cada Job para ser procesado, para esto se obtiene 1 registro de los Jobs que se encuentren en estado de 'NEW'.
- Una vez obtenido este registro se obtiene el tamaño del grupo para, obtener la cantidad de grupos que se deben realizar.
- Crear la cantidad de grupos que se necesitan.
- Actualizar en la tabla Jobs, el Job tomado debe tener su estado a 'IN-PROGRESS' y el campo Loader con el ID del pod que lo ha procesado.
- Enviar mensaje a una cola de RabbitMQ para que el siguiente componente reciba la información de los grupos que han sido procesados y creados.

Como prueba unitaria, se encuentra un archivo de prueba llamado `unitTest.py` en el directorio '`\Images\Loader\UnitTest`'

Este archivo funciona de manera similar al código de producción, sin embargo, contiene dos diferencias principales:

1. En esta prueba unitaria se tiene un ciclo FOR como principal para el procesamiento de los Jobs sea finito, pues el objetivo es ejemplificar el funcionamiento del mismo. El código de producción trabaja con un ciclo infinito de un WHILE True.
2. Esta prueba unitaria no crea las colas ni envía mensajes, pues todo lo que pasa lo muestra en consola.

Para ejecutar este archivo debe tener presente lo siguiente:

- Actualizar la contraseña de MariaDB
- Modificar variables de control como el sleeptime o la cantidad de Jobs para la prueba.

Primeramente, el programa inserta Jobs de prueba en una base de datos de prueba respectiva para su función y se obtienen datos similares a los siguientes:

id	created	status	end	loader	grp_size
1	2022-11-24 02:14:58	NEW	NULL	NULL	1338
2	2022-11-24 02:14:58	NEW	NULL	NULL	1121
3	2022-11-24 02:14:58	NEW	NULL	NULL	192
4	2022-11-24 02:14:58	NEW	NULL	NULL	1055
5	2022-11-24 02:14:58	NEW	NULL	NULL	217

Seguidamente el programa toma 1 Job para su procesamiento, el cual consiste en tomar el grp_size y el total de documentos del api de BioRxiv para saber la cantidad de grupos que se tienen que crear para ese Job en específico. A partir de este punto, se puede ver el funcionamiento del programa en la creación de los grupos, la información se muestra en consola de la siguiente manera:

```
OK: Connection successfully
Processing: Starting the processing of id_job 1
Processing: Group {'id_job': 1, 'grp_number': 0} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 1} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 2} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 3} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 4} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 5} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 6} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 7} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 8} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 9} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 10} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 11} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 12} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 13} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 14} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 15} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 16} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 17} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 18} has been processed successfully.
Processing: Group {'id_job': 1, 'grp_number': 19} has been processed successfully.
```

El programa ha tomado el Job con ID = 1 para ser procesado, pues es el primero que se ha encontrado con el estado de 'NEW', indicando que aún falta por ser procesado. Una vez tomado se procesan todos sus grupos, en el momento de la ejecución de esta prueba, el sitio de BioRxiv contiene 25609 registros, así que para la cantidad de grupos que se necesitan se obtiene con la división de 25609/1338 dando como resultado 19.14, lo que nos indica que se necesitan 20 grupos, pues se van a lograr realizar 19 grupos de 1338 documentos y los que sobran van en el último grupo que guarda los restantes, es por eso que va de 0 hasta 19 grupos.

El procesamiento se ve reflejado en MariaDB de la siguiente manera:

Journarl Search Platform

id	id_job	created	end	stage	grp_number	status	offset
1	1	2022-11-24 02:15:04	NULL	Loader	0	NULL	0
2	1	2022-11-24 02:15:04	NULL	Loader	1	NULL	1338
3	1	2022-11-24 02:15:04	NULL	Loader	2	NULL	2676
4	1	2022-11-24 02:15:04	NULL	Loader	3	NULL	4014
5	1	2022-11-24 02:15:04	NULL	Loader	4	NULL	5352
6	1	2022-11-24 02:15:04	NULL	Loader	5	NULL	6690
7	1	2022-11-24 02:15:04	NULL	Loader	6	NULL	8028
8	1	2022-11-24 02:15:04	NULL	Loader	7	NULL	9366
9	1	2022-11-24 02:15:04	NULL	Loader	8	NULL	10704
10	1	2022-11-24 02:15:04	NULL	Loader	9	NULL	12042
11	1	2022-11-24 02:15:04	NULL	Loader	10	NULL	13380
12	1	2022-11-24 02:15:04	NULL	Loader	11	NULL	14718
13	1	2022-11-24 02:15:04	NULL	Loader	12	NULL	16056
14	1	2022-11-24 02:15:04	NULL	Loader	13	NULL	17394
15	1	2022-11-24 02:15:04	NULL	Loader	14	NULL	18732
16	1	2022-11-24 02:15:04	NULL	Loader	15	NULL	20070
17	1	2022-11-24 02:15:04	NULL	Loader	16	NULL	21408
18	1	2022-11-24 02:15:04	NULL	Loader	17	NULL	22746
19	1	2022-11-24 02:15:04	NULL	Loader	18	NULL	24084
20	1	2022-11-24 02:15:04	NULL	Loader	19	NULL	25422

Estos han sido los grupos que se han insertado con los siguientes datos: * id_job: Job al que pertenece el grupo. * created: Indica el momento en que se creó el grupo. * end: momento de finalización. * stage: componente que lo está trabajando, en este caso, Loader. * grp_number: el número del grupo. * status: estado en que se encuentra el grupo, al ser la primera vez que se crea, este comienza en null. * offset: cantidad del desplazamiento sobre los documentos.

También, en el momento que se toma el Job para su procesamiento, se modifica su estado a 'In-Progress' y su campo 'loader' se actualiza con el nombre del pod que ha tomado ese job, en este caso es un nombre genérico al ser una prueba unitaria, por lo que después del procesamiento de cada uno de los distintos Jobs de prueba se obtiene una tabla similar a:

id	created	status	end	loader	grp_size
1	2022-11-24 02:14:58	In-progress	NULL	LOADER-UNITTEST	1338
2	2022-11-24 02:14:58	In-progress	NULL	LOADER-UNITTEST	1121
3	2022-11-24 02:14:58	In-progress	NULL	LOADER-UNITTEST	192
4	2022-11-24 02:14:58	In-progress	NULL	LOADER-UNITTEST	1055
5	2022-11-24 02:14:58	In-progress	NULL	LOADER-UNITTEST	217

Prueba unitaria del Downloader

Este componente se encarga de insertar datos en la tabla *history*, actualizar los campos determinados en la tabla *groups*, descargar los documentos del grupo indicado por el mensaje que recibe de la cola y almacenarlos en el índice de ElasticSearch *groups*.

Para la prueba unitaria se ejecuta el archivo de python *downloader_PruebaUnitaria.py* que se encuentra en:

```
pruebas/pruebaUnitaria_downloader/downloader_PruebaUnitaria.py
```

Este archivo publica cuatro mensajes a la cola que el downloader revisa con el formato

```
{
  "id_job": "{INT}",
  "grp_number": "{INT}"
}
```

Todos los mensajes poseen un id_job de 0 y el grp_number va desde el 0 hasta el número 3.

Además, para fines de la prueba se agregaron los records necesarios a las tablas *jobs* y *groups* desde MySQL Workbench.

	id	created	status	end	loader	grp_size
▶	0	2022-11-22 21:34:01.0	new	NULL	idloaderprueba	10
*	NULL	NULL	NULL	NULL	NULL	NULL

	id	id_job	created	end	stage	grp_number	status	offsett
▶	0	0	2022-11-22 21:30:58	NULL	loader	0	new	10
	1	0	2022-11-22 21:31:05	NULL	loader	1	new	10
	2	0	2022-11-22 21:31:10	NULL	loader	2	new	10
*	3	0	2022-11-22 21:31:15	NULL	loader	3	new	10
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

- Primero, cada 2 segundos se publica un mensaje, en la siguiente foto se puede observar que todos los mensajes han sido publicados. En el primer momento en que una de las réplicas del downloader recibe un mensaje inicia con sus tareas.

```
Simulation: Got msg from queue, start process
{
  "id_job": "0",
  "grp_number": "0"
}
Simulation: Got msg from queue, start process
{
  "id_job": "0",
  "grp_number": "1"
}
Simulation: Got msg from queue, start process
{
  "id_job": "0",
  "grp_number": "2"
}
Simulation: Got msg from queue, start process
{
  "id_job": "0",
  "grp_number": "3"
}
```

- En los logs de cada replica del componente Downloader, que se pueden visualizar desde el programa *lens*, se pueden apreciar las notificaciones que demuestran que las funciones se están ejecutando exitosamente.

Journarl Search Platform

```

Logs from 11/22/2022, 9:39:53 PM
 Show timestamps  Show previous terminated container 
```

Namespace	Owner	ReplicaSet	Pod	Container	Search...
defa...		downloader-6f6d5b696c-2t...	downloader-6f6d5b696c-2t...	downloader	<input type="text"/>

```

2022-11-22T21:39:53-06:00 Downloader: Message was received
2022-11-22T21:39:53-06:00 Downloader: Group table updated
2022-11-22T21:39:53-06:00 Downloader: Added record to history table
2022-11-22T21:40:21-06:00 Downloader: Published documents
2022-11-22T21:40:22-06:00 Downloader: Process finished
2022-11-22T21:40:22-06:00 Downloader: Message was received
2022-11-22T21:40:22-06:00 Downloader: Group table updated
2022-11-22T21:40:22-06:00 Downloader: Added record to history table
2022-11-22T21:40:25-06:00 Downloader: Published documents
2022-11-22T21:40:25-06:00 Downloader: Process finished

```



```

Logs from 11/22/2022, 9:39:53 PM
 Show timestamps  Show previous terminated container 
```

Namespace	Owner	ReplicaSet	Pod	Container	Search...
defa...		downloader-6f6d5b696c-jmcwb	downloader-6f6d5b696c-jm...	downloader	<input type="text"/>

```

2022-11-22T21:39:55-06:00 Downloader: Message was received
2022-11-22T21:39:55-06:00 Downloader: Group table updated
2022-11-22T21:39:55-06:00 Downloader: Added record to history table
2022-11-22T21:40:21-06:00 Downloader: Published documents
2022-11-22T21:40:22-06:00 Downloader: Process finished

```



```

Logs from 11/22/2022, 9:39:55 PM
 Show timestamps  Show previous terminated container 
```

Namespace	Owner	ReplicaSet	Pod	Container	Search...
defa...		downloader-6f6d5b696c-mqpf6	downloader-6f6d5b696c-mq...	downloader	<input type="text"/>

```

2022-11-22T21:39:51-06:00 Downloader: Message was received
2022-11-22T21:39:51-06:00 Downloader: Group table updated
2022-11-22T21:39:51-06:00 Downloader: Added record to history table
2022-11-22T21:40:21-06:00 Downloader: Published documents
2022-11-22T21:40:22-06:00 Downloader: Process finished

```

- Desde MySQL Workbench se puede consultar las tablas para verificar si los records se han añadido a la tabla *history* y si los campos de la tabla *groups* se han actualizado correctamente.
- Los campos *status*, *end* y *message* de la tabla *history* se han actualizado, al igual que el *status* del grupo en la tabla *groups*.

History

Journarl Search Platform

	id	component	status	created	end	message	stage	grp_id
▶	1	downloader-6f6d5b696c-mqpf6	completed	2022-11-22 21:39:51	2022-11-22 21:40:21	NULL	downloader	0
▶	2	downloader-6f6d5b696c-2tk6d	completed	2022-11-22 21:39:53	2022-11-22 21:40:21	NULL	downloader	1
▶	3	downloader-6f6d5b696c-jmcwb	completed	2022-11-22 21:39:55	2022-11-22 21:40:21	NULL	downloader	2
▶	4	downloader-6f6d5b696c-2tk6d	completed	2022-11-22 21:40:22	2022-11-22 21:40:25	NULL	downloader	3
*	HULL	NULL		NULL	NULL	NULL	NULL	NULL

Groups

	id	id_job	created	end	stage	grp_number	status	offsett
▶	0	0	2022-11-22 21:30:58	NULL	downloader	0	completed	10
▶	1	0	2022-11-22 21:31:05	NULL	downloader	1	completed	10
▶	2	0	2022-11-22 21:31:10	NULL	downloader	2	completed	10
▶	3	0	2022-11-22 21:31:15	NULL	downloader	3	completed	10
*	HULL	NULL	NULL	NULL	NULL	HULL	NULL	NULL

- Desde RabbitMQ se puede observar que los cuatro mensajes por parte del Downloader han sido publicados, listos para ser consumidos por el siguiente componente.

RabbitMQ™ RabbitMQ 3.10.7 Erlang 24.3.4

Refreshed 2022-11-22 21:42:26 Refresh every 5 seconds

Virtual host All User user Log out

Overview Connections Channels Exchanges Queues Admin

Cluster rabbit@databases-rabbitmq-0.databases-rabbitmq-headless.default.svc.cluster.local

Queues

All queues (2)

Pagination

Page 1 of 1 - Filter: Regex ?

Displaying 2 items , page size up to: 100

Overview		Messages			Message rates			+/-	
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
downloader	classic		idle	4	0	4	0.00/s		
loader	classic		idle	0	0	0	0.00/s	0.00/s	0.00/s

Add a new queue

HTTP API Server Docs Tutorials Community Support Community Slack Commercial Support Plugins GitHub Changelog

- Por último, desde ElasticSearch se puede comprobar que los mensajes han sido almacenados existosamente en el índice correspondiente. En la primera imagen se puede ver el texto resaltado, que indica que se encontraron 4 elementos en el índice. En las siguientes imágenes se puede observar que lo que se publicó a ElasticSearch posee el formato establecido por el grupo al igual que el campo de docs con los 10 documentos del grupo.

```

1 GET groups/_search ➔ 🔍
2 {
3   "size":500
4 }
5
6 DELETE groups
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
    "took": 418,
    "timed_out": false,
    "_shards": {
      "total": 1,
      "successful": 1,
      "skipped": 0,
      "failed": 0
    },
    "hits": {
      "total": {
        "value": 4,
        "relation": "eq"
      },
      "max_score": 1,
      "hits": [
        {
          "_index": "groups",
          "_id": "museooQB2NgTtqJpQLGu",
          "_score": 1,
          "ignored": [
            "docs.rel_abs.keyword"
          ],
          "_source": {
            "id_job": "0",
            "grp_number": "0",
            "docs": [
              {
                "rel_doi": "10.1101/2022.11.20.22282552",
                "rel_title": "Use of whole genome sequencing to identify low-frequency mutations in COVID-19 patients treated with remdesivir",
                "rel_date": "2022-11-22"
              }
            ]
          }
        }
      ]
    }
  }
}

```

Journarl Search Platform

```
24 |     "_source": {
25 |         "id_job": "0",
26 |         "grp_number": "0",
27 |         "docs": [
28 |             {
29 |                 "rel_doi": "10.1101/2022.11.20.22282552",
30 |                 "rel_title": "Use of whole genome sequencing to identify low-frequency mutations in COVID-19 patients treated with remdesivir",
31 |                 "rel_date": "2022-11-22",
32 |                 "rel_site": "medRxiv",
33 |                 "rel_link": "https://medrxiv.org/cgi/content/short/2022.11.20.22282552",
34 |                 "rel_abs": "Background: We investigate the effects of remdesivir (RDV) treatment on intra-host SARS-CoV-2 diversity and low -frequency mutations in moderately ill hospitalized COVID-19 patients and compare them to patients without RDV treatment. Methods: Sequential collections of nasopharyngeal and mid-turbinate swabs were obtained from 16 patients with and 31 patients without RDV treatment. A total of 113 samples were sequenced and mutation analyses were performed. Results: We did not identify any drug resistant mutations during RDV therapy. In genes encoding and associated with the replication complex, low -frequency minority variants that do not reach fixation within the sampling period were detected in 6/16 (37.5%) and 14/31 (45%) patients with and without RDV treatment respectively. We did not detect significant differences in within-host diversity and positive selection between the RDV-treated and untreated groups. Conclusions: Minimal intra-host variability and stochastic low-frequency variants detected in moderately ill patients suggests little selective pressure in patients receiving short courses of RDV. Patients undergoing short regimens of RDV therapy should continue to be monitored.",
35 |                 "rel_num_authors": 11,
36 |                 "rel_authors": [
37 |                     {
38 |                         "author_name": "Kuganya Nirmalarajah",
39 |                         "author_inst": "Sunnybrook Research Institute"
40 |                     },
41 |                     {
42 |                         "author_name": "Finlav Maguire".
43 |                     }
44 |                 ],
45 |                 "_index": "groups",
46 |                 "_id": "n0eSooQB2NgTtqJpS7Hu",
47 |                 "score": 1,
48 |                 "_ignored": [
49 |                     "docs.rel_abs.keyword"
50 |                 ],
51 |                 "source": {
52 |                     "id_job": "0",
53 |                     "grp_number": "1",
54 |                     "docs": [
55 |                         {
56 |                             "rel_doi": "10.1101/2022.11.22.517073",
57 |                             "rel_title": "Environmental and genetic drivers of population differences in SARS-CoV-2 immune responses",
58 |                             "rel_date": "2022-11-22",
59 |                             "rel_site": "bioRxiv",
60 |                             "rel_link": "https://biorkxiv.org/cgi/content/short/2022.11.22.517073",
61 |                             "rel_abs": "Humans display vast clinical variability upon SARS-CoV-2 infection, partly due to genetic and immunological factors However, the magnitude of population differences in immune responses to SARS-CoV-2 and the mechanisms underlying such variation remain unknown. Here we report single-cell RNA-sequencing data for peripheral blood mononuclear cells from 222 healthy donors of various ancestries stimulated with SARS-CoV-2 or influenza A virus. We show that SARS-CoV-2 induces a weak , but more heterogeneous interferon-stimulated gene activity than influenza A virus, and a unique pro-inflammatory signature in myeloid cells. We observe marked population differences in transcriptional responses to viral exposure that reflect environmentally induced cellular heterogeneity, as illustrated by higher rates of cytomegalovirus infection, affecting lymphoid cells, in African-descent individuals. Expression quantitative trait loci and mediation analyses reveal a broad effect of cell proportions on population differences in immune responses, with genetic variants having a narrower but stronger effect on specific loci. Additionally, natural selection has increased immune response differentiation across populations, particularly for variants associated with SARS-CoV-2 responses in East Asians. We document the cellular and molecular
62 |                         ],
63 |                     ]
64 |                 ],
65 |                 "id_job": "0",
66 |                 "grp_number": "2",
67 |                 "docs": [
68 |                     {
69 |                         "rel_doi": "10.1101/2022.11.20.517236",
70 |                         "rel_title": "Polymorphic regions in BA.2.12.1, BA.4 and BA.5 likely implicated in immunological evasion of Omicron subvariant BQ.1.1",
71 |                         "rel_date": "2022-11-21",
72 |                         "rel_site": "bioRxiv",
73 |                         "rel_link": "https://biorkxiv.org/cgi/content/short/2022.11.20.517236",
74 |                         "rel_abs": "In this work, 45 Spike glycoprotein Chain B polypeptides were used in the subvariants BA.2.12.1, BA.4 and BA.5 were recovered from GENBANK. All sequences were publicly available on the National Biotechnology Information Center (NCBI) platform . The results indicate the existence of informative polymorphic and parsimony sites that may be implicated in the level of diversity of the studied strains, as well as reflect the immunological evasion potential of the subvariant BQ1.1. of the variant Omicron d and SARS-CoV-2. The results also suggest the formation of ancestral polymorphism with slight retention, and the probable is responsible the diversity of the whole studied set.",
75 |                         "rel_num_authors": 1,
76 |                         "rel_authors": [
77 |                             {
78 |                                 "author_name": "Pierre Teodosio Felix Sr.",
79 |                                 "author_inst": "Laboratory of Population Genetics and Computational Evolutionary Biology - LaBECom, UNIVISA, Vitoria de Santo Antao, Pernambuco, Brazil."
80 |                             }
81 |                         ],
82 |                         "version": "1",
83 |                         "license": "cc hv nc".
84 |                     }
85 |                 ],
86 |             ]
87 |         ]
88 |     ]
89 | 
```

Journal Search Platform

```
1085 },
1737 {
1738     "_index": "groups",
1739     "_id": "neeSooQ82NgTtqJpj7GA",
1740     "_score": 1,
1741     "_ignored": [
1742         "docs.rel_abs.keyword"
1743     ],
1744     "_source": {
1745         "id_job": "0",
1746         "grp_number": "3",
1747         "docs": [
1748             {
1749                 "rel_doi": "10.1101/2022.11.18.22282414",
1750                 "rel_title": "Immunogenicity and safety of a 4th homologous booster dose of a SARS-CoV-2 recombinant spike protein vaccine (NVX-CoV2373): a phase 2, randomized, placebo-controlled trial",
1751                 "rel_date": "2022-11-20",
1752                 "rel_site": "medRxiv",
1753                 "rel_link": "https://medrxiv.org/cgi/content/short/2022.11.18.22282414",
1754                 "rel_abs": "BackgroundThe emergence of SARS-CoV-2 variants has significantly reduced the efficacy of some approved vaccines. A fourth dose of NVX-CoV2373 (5{micro}g SARS-CoV-2 rs + 50{micro}g Matrix-M adjuvant) was evaluated to determine induction of cross-reactive antibodies to variants of concern."
1755
1756 MethodsA phase 2 randomized study assessed a fourth dose of NVX-CoV2373 in adults 18-84 years of age (2-dose primary series followed by third and fourth doses at 6-month intervals). Local/systemic reactogenicity was assessed the day of vaccination and for 6 days thereafter. Unsolicited adverse events (AEs) were reported. Immunogenicity was measured before, and 14 days after, fourth dose administration using anti-spike neutralization assays against the ancestral SARS-CoV-2 strain and Omicron sublineages. Antigenic cartography assessed antigenic distances between ancestral and variant strains.
1757
1758 ResultsAmong 1282 enrolled participants, 258 were randomized to receive the 2-dose primary series, of whom 101 received a third dose, and 15
```

Pruebas Unitarias de Details Downloader

El Details Downloader se va a encargar de tomar el rel_doi de cada documento de elasticsearch para buscar los detalles por medio del api si es que estos existen. Por último, le va a agregar los detalles al documento. Ubicación del script de prueba: \\Images\\Details Downloader\\UnitTest\\test.json Ubicación del JSON de prueba: \\Images\\Details Downloader\\UnitTest\\prueba.json

Así es como se ve uno de los documentos de prueba antes de ser procesado

```
{
  "rel_doi": "10.1101/2022.11.09.22282113",
  "rel_title": "Risk of SARS-CoV-2 reinfection is time- and variant-dependant, France, January 2021 to August 2022",
  "rel_date": "2022-11-15",
  "rel_site": "medRxiv",
  "rel_link": "https://medrxiv.org/cgi/content/short/2022.11.09.22282113",
  "rel_abs": "Since the emergence of Omicron, reinfections with SARS-CoV-2 have been rising. We estimated the risk of",
  "rel_num_authors": 7,
  "rel_authors": [
    {
      "author_name": "Vincent Auvigne",
      "author_inst": "Sante Publique France"
    },
    {
      "author_name": "Justine Schaeffer",
      "author_inst": "Sante publique France"
    },
    {
      "author_name": "Thibault Boudon",
      "author_inst": "Sante publique France"
    },
    {
      "author_name": "Cynthia Tamandjou",
      "author_inst": "Sante publique France"
    },
    {
      "author_name": "Julie Figoni",
      "author_inst": "Sante publique France"
    },
    {
      "author_name": "Isabelle Parent du Chatelet",
      "author_inst": "Sante publique France"
    },
    {
      "author_name": "Sibylle Bernard-Stoecklin",
      "author_inst": "Sante publique France"
    }
  ],
  "version": "1",
  "license": "cc_by",
  "type": "PUBLISHAHEADOFPRINT",
  "category": "epidemiology"
},
}
```

Luego cambiamos los valores de: ELASTICPASS, MARIADBPASS, y RABBITPASS por los que corresponden a los passwords generados para los deployments actuales y corremos el script de prueba. Cuando revisamos el índice, podemos ver que se ha agregado el field details.

Journarl Search Platform

```
[{"rel_doi": "10.1101/2022.11.09.22282113",
"rel_title": "Risk of SARS-CoV-2 reinfection is time- and variant-dependant, France, January 2021 to August 2022",
"rel_date": "2022-11-15",
"rel_site": "medRxiv",
"rel_link": "https://medrxiv.org/cgi/content/short/2022.11.09.22282113",
"rel_abs": "Since the emergence of Omicron, reinfections with SARS-CoV-2 have been rising. We estimated the risk of SARS-CoV-2 reinfection with pre-Delta variants being the reference group, was estimated at 0.43 [95%CI 0.40-0.47] if the primary infection was BA.2 or BA.4/5 reinfection. If the primary infection the protection was similar against BA.2 or BA.4/5 reinfection.",
"rel_num_authors": 7,
"rel_authors": [
  {
    "author_name": "Vincent Auvigne",
    "author_inst": "Sante Publique France"
  },
  {
    "author_name": "Justine Schaeffer",
    "author_inst": "Sante publique France"
  },
  {
    "author_name": "Thibault Boudon",
    "author_inst": "Sante publique France"
  },
  {
    "author_name": "Cynthia Tamandjou",
    "author_inst": "Sante publique France"
  },
  {
    "author_name": "Julie Figoni",
    "author_inst": "Sante publique France"
  },
  {
    "author_name": "Isabelle Parent du Chatelet",
    "author_inst": "Sante publique France"
  },
  {
    "author_name": "Sibylle Bernard-Stoecklin",
    "author_inst": "Sante publique France"
  }
],
"version": "1",
"license": "cc_by",
"type": "PUBLISHAHEADOFPRINT",
"category": "epidemiology",
"details": {
  "doi": "10.1101/2022.11.09.22282113",
  "title": "Risk of SARS-CoV-2 reinfection is time- and variant-dependant, France, January 2021 to August 2022",
  "authors": "Auvigne, V.; Schaeffer, J.; Boudon, T.; Tamandjou, C.; Figoni, J.; Parent du Chatelet, I.; Bernard-Stoecklin, S.",
  "author_corresponding": "Vincent Auvigne",
  "author_corresponding_institution": "Sante Publique France",
  "date": "2022-11-15",
  "version": "1",
  "type": "PUBLISHAHEADOFPRINT",
  "license": "cc_by",
  "category": "epidemiology",
  "jatsxml": "https://www.medrxiv.org/content/early/2022/11/15/2022.11.09.22282113.source.xml",
  "abstract": "Since the emergence of Omicron, reinfections with SARS-CoV-2 have been rising. We estimated the risk of SARS-CoV-2 reinfection with pre-Delta variants being the reference group, was estimated at 0.43 [95%CI 0.40-0.47] if the primary infection was BA.2 or BA.4/5 reinfection. If the primary infection the protection was similar against BA.2 or BA.4/5 reinfection.",
  "published": "NA",
  "server": "medrxiv"
}
},
{
  "details-downloader-67cd4dbbf-phd49", 'Completed', datetime.datetime(2022, 11, 24, 2, 10, 39), datetime.datetime(2022, 11, 24, 2, 10, 46), 'succesful process', 232, 'details-downloader')
}
```

En MariaDB podemos ver que la tabla history tiene la entrada completada

```
(1, 'details-downloader-67cd4dbbf-phd49', 'Completed', datetime.datetime(2022, 11, 24, 2, 10, 39), datetime.datetime(2022, 11, 24, 2, 10, 46), 'succesful process', 232, 'details-downloader'))
```

También podemos ver el cambio en la tabla groups

```
[(232, 1, datetime.datetime(2022, 11, 23, 21, 43, 58), None, 'details-downloader', 3, 'Completed', 20)]
```

Pruebas Unitarias de Jatsxml Processor

Journarl Search Platform

Para ejecutar la prueba unitaria del Jatsxml Processor se necesita de dos cosas:

- Instalar las bases de datos y RabbitMQ
- Instalar el componente
- Cambiar las credenciales del script de prueba
- Ejecutar el script de prueba

Las credenciales del script se cambian como dice en la sección de [contraseña](#). El script de prueba se encuentra en la carpeta: *Images->Jatsxml Processor->UnitTest->test.py*

Para la prueba se procesará un "group" previamente creado con 5 documentos donde cada uno tiene el parámetro jatsxml en details, este "group" se encuentra en: *Images->Jatsxml Processor->UnitTest->prueba.json*

Al ejecutar la prueba la cola de donde consume el componente se le manda el siguiente mensaje:

```
{
  "id_job": "1",
  "grp_number": "3"
}
```

Overview				Messages			Message rates		
Name	Type	Features	State	Ready	Unacked	Total	incoming	deliver / get	ack
details-downloader	classic		idle	1	0	1	0.00/s	0.20/s	0.00/s

Se crea un group y un job en mariaDB:

	id	id_job	created	end	stage	grp_number	status	offset
▶	232	1	2022-11-24 20:38:09	NULL	jatsxml-processor	3	In-progress	20
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

	id	created	status	end	loader	grp_size
▶	1	2022-11-24 20:38:09	new	NULL	loader_23i0sk2	5
*	NULL	NULL	NULL	NULL	NULL	NULL

El grupo [prueba.json](#) se introduce en elasticsearch en el índice groups:

```

18 |     "_index": "groups",
19 |     "_id": "jRerrIQBGVxNktSysx7i",
20 |     "_score": 1,
21 |     "_ignored": [
22 |       "docs.rel_abs.keyword"
23 |     ],
24 |     "_source": {
25 |       "id_job": "1",
26 |       "grp_number": "3",
27 |       "docs": [
28 |         {
29 |           "rel_doi": "10.1101/2022.11.12.22282248",
30 |           "rel_title": "COVID-19 IN BRAZIL: A CROSS-SECTIONAL IMMUNO-EPIDEMICAL AND GENOMIC EVALUATION IN A
31 |             PRE-OMICRON ERA",
32 |           "rel_date": "2022-11-15",
33 |           "rel_site": "medRxiv",
34 |           "rel_link": "https://medrxiv.org/cgi/content/short/2022.11.12.22282248",
35 |           "rel_abs": "The seventh human coronavirus, was discovered and reported primarily in Wuhan, China.
36 |             After intense seasons with repercussions in all areas of humanity, the pandemic demonstrates a
37 |               new perspective. In Brazil, the pandemic concept has had impacts in vast areas, including
38 |                 mainly hospitals. This present study aims to describe, present and synthesize data that
39 |                   correlate the symptoms of passive and/or active patients for Covid-19 and their respective
40 |                     results of IgG/IgM serological tests in hospitals in Cruzeiro, São Paulo. The form had been
41 |                       applied to 333 people and obtained conclusive results and several symptoms presented, in
42 |                         addition, asymptomatic cases were also analyzed and directed in the genomic study of variants
43 |                           of concern, as well as vaccination data in the study region.",
44 |           "rel_num_authors": 4,
45 |           "rel_authors": [
46 |             {
47 |               "author_name": "Sarah de Oliveira Rodrigues",
48 |             }
49 |           ]
50 |         }
51 |       ]
52 |     }
53 |
54 |   }
55 |
56 | 
```

Resultados:

Los logs que se reflejan en el componente son los siguientes:

```

Message Receive: Starting Process -> {'id_job': '1', 'grp_number': '3'}
Processing: success at updating group table -> grp_number: 3 id_job: 1
Processing: success at creating new registry in history table -> grp_number: 3 id_job: 1
Processing: success at getting group from elastic -> grp_number: 3 id_job: 1
Processing: success at getting Jatsxml -> grp_number: 3 id_job: 1
Processing: success at publishing new doc -> grp_number: 3 id_job: 1
Processing: success at getting Jatsxml -> grp_number: 3 id_job: 1
Processing: success at publishing new doc -> grp_number: 3 id_job: 1
Processing: success at getting Jatsxml -> grp_number: 3 id_job: 1
Processing: success at publishing new doc -> grp_number: 3 id_job: 1
Processing: success at getting Jatsxml -> grp_number: 3 id_job: 1
Processing: success at publishing new doc -> grp_number: 3 id_job: 1
Processing: success at getting Jatsxml -> grp_number: 3 id_job: 1
Processing: success at publishing new doc -> grp_number: 3 id_job: 1
Processing: success at deleting group -> grp_number: 3 id_job: 1
Processing: success at updating group table -> grp_number: 3 id_job: 1
Processing: success at updating jobs table -> grp_number: 3 id_job: 1
Processing: success at modifying history -> grp_number: 3 id_job: 1
Group finished: -> grp_number: 3 id_job: 1

```

En elasticsearch, el indice "registries" que es el nombre puesto en el values.yaml del App se publican 5 documentos:

```
"hits": {
  "total": {
    "value": 5,
    "relation": "eq"
  },
  "max_score": 1,
  "hits": [
    {
      "_index": "registries",
      "_id": "rQWtrIQBus4QMSxCj-39",
      "_score": 1,
      "_ignored": [
        "rel_abs.keyword"
      ],
      "_source": {
        "rel_doi": "10.1101/2022.11.12.22282248",
        "rel_title": "COVID-19 IN BRAZIL: A CROSS-SECTIONAL IMMUNO-EPIDEMICAL AND GENOMIC EVALUATION IN A PRE-OMICRON ERA",
        "rel_date": "2022-11-15",
        "rel_site": "medRxiv",
        "rel_link": "https://medrxiv.org/cgi/content/short/2022.11.12.22282248",
        "rel_abs": "The seventh human coronavirus, was discovered and reported primarily in Wuhan, China. After intense seasons with repercussions in all areas of humanity, the pandemic demonstrates a new perspective. In Brazil, the pandemic concept has had impacts in vast areas, including mainly hospitals. This present study aims to describe, present and synthesize data that correlate the symptoms of passive and/or active patients for Covid-19 and their respective results of IgG/IgM serological tests in hospitals in Cruzeiro, São Paulo. The form had been applied to 333 people and obtained conclusive results and several symptoms presented, in addition, asymptomatic cases were also analyzed and directed in the genomic study of variants of concern, as well as vaccination data in the study region.",
        "rel_num_authors": 4,
        "rel_authors": [
          {
            "author_name": "Sarah de Oliveira Rodrigues",
            "author_inst": "Pontifical Catholic University"
          }
        ]
      }
    }
  ]
}
```

Los documentos tienen jatsxml:

Journarl Search Platform

```

"type": "PUBLISHERABOUTPRINT",
"category": "epidemiology",
"details": {
  "jatsxml": {
    "article": {
      "@article-type": "article",
      "@specific-use": "production",
      "@xml:lang": "en",
      "@xmlns:hw": "org.highwire.hpp",
      "@xmlns:mml": "http://www.w3.org/1998/Math/MathML",
      "@xmlns:xsi": "http://www.w3.org/2001/XMLSchema-instance",
      "@xmlns:ali": "http://www.niso.org/schemas/ali/1.0/",
      "@xmlns:ref": "http://schema.highwire.org/Reference",
      "@xmlns:hwp": "http://schema.highwire.org/Journal",
      "@xmlns:l": "http://schema.highwire.org/Linking",
      "@xmlns:r": "http://schema.highwire.org/Revision",
      "@xmlns:x": "http://www.w3.org/1999/xhtml",
      "@xmlns:app": "http://www.w3.org/2007/app",
      "@xmlns:xlink": "http://www.w3.org/1999/xlink",
      "@xmlns:nlm": "http://schema.highwire.org/NLM/Journal",
      "@xmlns:a": "http://www.w3.org/2005/Atom",
      "@xmlns:c": "http://schema.highwire.org/Compound",
      "@xmlns:_hpp": "http://schema.highwire.org/Publishing",
      "front": {
        "journal-meta": {
          "journal-id": [
            {
              "@journal-id-type": "hwp",
              "#text": "medrxiv"
            },
            {
              "@journal-id-type": "publisher-id",
              "#text": "MEDRXIV"
            }
          ]
        }
      }
    }
  }
}

```

El jobs se actualiza:

	id	created	status	end	loader	grp_size
▶	1	2022-11-24 20:44:05	Completed	2022-11-24 20:46:14	loader_23i0sk2	5
*	NULL	NULL	NULL	NULL	NULL	NULL

La tabla history se actualiza:

	id	component	status	created	end	message	grp_id	stage
▶	5386	jatsxml-processor-c594ff4f5-w4gvt	Completed	2022-11-24 20:46:05	2022-11-24 20:46:14	successful process	232	jatsxml-processor
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

La tabla groups se actualiza:

	id	id_job	created	end	stage	grp_number	status	offset
▶	232	1	2022-11-24 20:44:05	2022-11-24 20:46:14	jatsxml-processor	3	Completed	20
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Prueba unitaria del API

Para ejecutar estas pruebas se necesitan "unitTest.py" y "script.py" localizadas en el branch del Api, además tener en cuenta que se trabajarán de manera local y controlada.

Para iniciar se debe ejecutar el archivo script.py y darle a la opción "4"

- Crea una database en MariaDB, la cual se llamará "pruebaUnitaria"
- Crea un indice llamado "articulos" y hace un mapping a un field en específico, para que no existan problemas a la hora de realizar el search en Elastic.
- Por último, inserta 200 artículos en el indice creado anteriormente.

Antes de ejecutar todas las pruebas se debe actualizar todas las credenciales de las bases de datos.

```
MariaClient = mariadb.connect(  
    host='localhost',  
    port= 32100,  
    user='root',  
    password= "SlvQ0xsghQ",  
    database= 'pruebaUnitaria')  
cur = MariaClient.cursor()  
  
ElasticClient = Elasticsearch(  
    "http://localhost:53608/",  
    basic_auth=("elastic","Gu6cigeirVvLXZAa"))  
)
```

El archivo "unitTest.py" está conformado por 12 funciones, cada una dividida de tal forma que se pueda mostrar los casos en donde la ejecución es exitosa como no, ambas se diferencian al final del nombre con un OK o un FAIL.

```

def test_insertar_job_ok(self): ...

def test_insertar_job_FAIL(self): ...

def test_buscar_OK(self): ...

def test_buscar_FAIL(self): ...

def test_detalles_OK(self): ...

def test_detalles_FAIL(self): ...

def test_detalles_like_OK(self): ...

def test_detalles_like_FAIL(self): ...

def test_like_OK(self): ...

def test_like_FAIL(self): ...

def test_listalikes_ok(self): ...

def test_listalikes_FAIL(self): ...

```

- Test insertar un job:** Lo que hace es insertar un job en MariaDB, específicamente en la base de datos creada en el script, la primera prueba se hace de manera correcta en donde la petición envía un número para dividir los grupos. En la segunda opción, se envía la petición vacía la cual provoca un error en la inserción.
- Test buscar artículos:** Hace un search en Elastic, en donde su query es la petición indicada al inicio de cada función. Para la primera prueba, se manda una petición que devuelve una respuesta este caso se toma como un OK, en el segundo caso se manda una palabra desconocida lo que provoca que no devuelva ningún artículo, este caso se toma como un FAIL.
- Test detalles de un artículo:** Busca el match con el título del artículo seleccionado y despliega la información. En el caso OK se toma un título existente el cual devolvió el search de elastic, para el caso FAIL se toma un título inexistente en la lista devuelta por Elastic.
- Test like:** Lo que hace es guardar en firebase el artículo al cual se le dio like. En el caso correcto, se selecciona un artículo que anteriormente no estaba guardado en la lista de likes en firebase, el caso FAIL se toma un artículo que anteriormente ya estaba guardado por lo que se toma como un error a la hora de guardarlo.
- Test lista de likes:** Despliega la lista de los artículos guardados en firebase, según uid del usuario. Se toma como OK cuando se despliega una lista de artículos y el uid del usuario es correcto. El caso FAIL se toma cuando el uid no tiene una lista de likes y no se despliega nada.
- Test detalles like:** Despliega la información del artículo guardado en firebase, en la lista de likes del usuario. Se toma como OK cuando el título hace match y se abre la información del artículo. El caso FAIL sería el título indicado no hace match.

Resultado de la prueba:

```
....  
....  
-----  
Ran 12 tests in 1.159s  
  
OK
```

Recomendaciones.

1. Si se utiliza el navegador de internet Chrome o Microsoft Edge, se recomienda instalar la extensión [JSON Formatter](#) para poder visualizar el API de BioRxiv formateado de forma que se tiene más legibilidad y mejor navegabilidad para leer el JSON.
2. Se recomienda mantener las variables con las credenciales locales utilizadas para conectarse a ElasticSearch, MariaDB y RabbitMQ en el código fuente de los componentes, de forma comentada, con el fin de tener fácil acceso a ellas en caso de que sea necesario realizar pruebas locales.
3. Buen conocimiento con transacciones dentro de bases de datos SQL para el uso de varias replicas y buena concurrencia.
Esto pues al momento de tener varias réplicas del componente de Loader, puede darse la situación donde varios pods tomen el mismo job, procesándolo varias veces.
4. El uso de contadores es algo simple pero funcional para las métricas, dentro de la librería de `prometheus_client` se pueden encontrar, esto es porque al momento de procesar X tarea, simplemente el contador se aumenta sin tanta complicación.
5. El uso de Summary nos ayuda en gran medida con la medición de tiempos de procesamiento por parte de cualquier componente que lo necesita, pues este nos da un recuento del tiempo que se ha tardado en X proceso y la cantidad de veces que se ha realizado.
6. Cuando se trabaja con Thunkable es importante entender el manejo y creación de las funciones a la hora de programar con bloques, pues facilita mucho tener un orden en el código y ejecutar las funcionalidades de la mejor manera.
7. Al trabajar con ngrok, pueden existir conflictos con solicitudes http y demás. En estos casos, lo ideal es utilizar el Intercambio de Recursos de Origen Cruzado (CORS) en Python.
8. Se recomienda a nuevos integrantes del proyecto que no conocen el funcionamiento de este, consultar a los miembros del equipo cómo funciona para poder hacer pruebas e integrar sus partes sin problemas.
9. Se recomienda en caso de tener suficientes recursos, remover las limitaciones de recursos establecidas a los deployments.
En este caso se usaron porque sin estas no nos corría.
10. Se recomienda crear una estructura inicial en el repositorio antes de iniciar cualquier proyecto, de esta forma las branches que se creen del main, pueden ser juntadas fácilmente sin muchos conflictos

Conclusiones.

1. La práctica del desarrollo de pipelines de un RestAPI es muy relevante hoy en día porque son utilizados en aplicaciones populares que implementan bases de datos en la nube como *Instagram*, *Telegram* y *Facebook*. Además, existe gran cantidad de páginas web que implementan los RestAPI, por lo que el conocimiento de este tipo de interfaces es de gran valor para los integrantes del equipo con interés en esta área.

2. Elasticsearch es una base de datos NoSQL que resulta conveniente para el caso de uso del presente proyecto, pues se encuentra optimizado para realizar búsquedas de texto, entonces es más beneficioso sobre otras bases de datos como MongoDB; dado que lo que se busca es realizar búsquedas de artículos científicos, lo que implica realizar búsquedas de texto.
3. El uso de bases de datos SQL es bastante funcional en el caso de tener alta concurrencia y tener control sobre la consistencia de los datos y saber que cada Job ha sido procesado solo una vez por un componente. A su vez tener un historial de trabajo para tener gran trazabilidad sobre las cosas que han sucedido.
4. Firebase es una base de datos muy simple de utilizar, especialmente cuando se trabaja con Thunkable esto se debe a que se puede conectar de una manera muy simple y su sistema de autenticación es eficiente.
5. Thunkable es una herramienta sencilla pero también es una plataforma que brinda muchas funcionalidades y robustez. Especialmente al desplegar listas y el manejo de las visibilidades en los elementos del diseño.
6. El uso de réplicas permite acelerar procesos en los que se deben procesar muchos documentos como el que se realiza en este proyecto.
7. Para busquedas en elasticsearch, si solo se va a usar una parte del o los documentos que se estan buscando, es buena idea usar el parametro de "source" y pasar una lista con los datos que se necesitan, de esta forma es mas rapida la consulta.
8. Se puede desactivar los mappings dinamicos de elasticsearch de forma parcial con datos que son muy poco predecibles y no tienen un patron. Esto debido a que elasticsearch intentara asignar un tipo de datos para indexar y si despues el tipo de dato era otro va a causar errores.
9. Parte de lo dicho anteriormente se puede evitar colocando en mappings la opcion llamada 'ignore_malformed' en True, lo que hara que se salte la indexacion de espacios que no sigan cierto patron (solo sirve con ciertos tipos de datos)
10. Siempre es bueno verificar incompatibilidades entre librerias. En este caso, Flask con la opcion de "debug" en true, no permite crear el servidor de metricas de prometheus.

Anexo 1: Referencias de Internet

Como es bien sabido, no somos conocedores de todas las soluciones ante todos los problemas, es por eso por lo que internet es una gran ayuda ante momentos donde necesitamos de algún tipo de apoyo en alguna sección de nuestro código. En este apartado veremos las secciones de este proyecto dónde se ha utilizado código fuera de nuestra creación, el funcionamiento y la razón de por qué han sido tomados.

Clase bcolors

Clase que modifica el color de la letra al momento de las impresiones en pantalla.

Código en el Programa

```
#Original idea was taken from: https://www.delftstack.com/es/howto/python/python-print-colored-text/
#and added some extra colors from: https://www.geeksforgeeks.org/print-colors-python-terminal/
You, hace 16 horas | 1 author (You)
class bcolors:
    OK      = '\033[92m'      #GREEN
    WARNING = '\033[93m'      #YELLOW
    FAIL    = '\033[91m'      #RED
    RESET   = '\033[0m'       #RESET COLOR
    black   = '\033[30m'
    red     = '\033[31m'
    green   = '\033[32m'
    orange  = '\033[33m'
    blue    = '\033[34m'
    purple  = '\033[35m'
    cyan    = '\033[36m'
    lightgrey= '\033[37m'
    darkgrey = '\033[90m'
    lightred = '\033[91m'
    lightgreen= '\033[92m'
    yellow  = '\033[93m'
    lightblue = '\033[94m'
    pink    = '\033[95m'
    lightcyan= '\033[96m'
```

Funcionamiento

Esta clase nos ayuda para que al momento de realizar impresiones dentro del programa, esto se logra escribiendo el color que necesitamos mostrar y el mensaje que se mostrara, después de haber realizado esto se debe reiniciar el color, esto ya que si no se hace todas las demás impresiones seguirán del mismo color seleccionado. La clase original solamente contenía los color correspondidos a *[OK,WARNING,FAIL,RESET]*, sin embargo al momento de las pruebas unitarias se notó que hacían falta más colores, es por eso que la clase original ha sido modificada para tener mayor cantidad de opciones.

Un ejemplo práctico de su uso es:

```
print(f"{bcolors.OK} REGEX PROCESSOR: {bcolors.RESET} Process Finished")
```

Razón de Uso

La razón de su elección es para una mejor legibilidad de mensajes tipo logs que se verán sobre el pod en el momento que la aplicación se encuentre funcionando, pues ver texto plano es más difícil de leer, así que usar esta clase nos hace mas sencillo el proceso de lectura por parte del usuario para saber que esta haciendo el pod en cada momento.

Referencias bibliográficas

DelftStack. (17 de diciembre del 2020). *Texto de color impreso en Python*. Delft Stack.

<https://www.delftstack.com/es/howto/python/python-print-colored-text/>

GeeksforGeeks. (27 de junio del 2022). *Print Colors in Python terminal*. <https://www.geeksforgeeks.org/print-colors-python-terminal/>