



BBS71



BLACK BIRD STACK



Integrantes:

Kevin Artunduaga Vivas - Angie Manzano Melendez
Samuel Pinzón Valderruten - Sebastian Díaz Noguera
Juan Camilo Vargas Vélez

BOARDING PASS

● FLIGHT

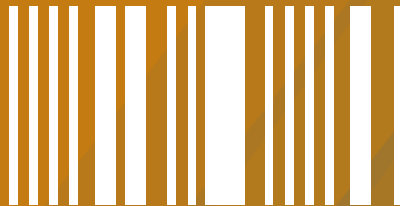
B345

● GATE

D8

● SEAT

29E





BBS71



Airline



Problema central

"El impacto negativo de los frecuentes retrasos de vuelos en la satisfacción de los pasajeros y la rentabilidad de las aerolíneas y aeropuertos como un problema crónico de la industria de viajes"



BBS71



Selección del dataset

kaggle



Este dataset fue hecho con el fin de predecir qué vuelos serán cancelados o retrasados y el tiempo de retraso.



61 Columnas



25.78 GB



ROB MULLA · UPDATED 6 MONTHS AGO



141

New Notebook



Download (4 GB)



Flight Status Prediction

Can you predict which flights will be delayed or cancelled in 5 years of data?





BBS71



Dataset usado

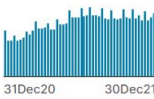

Debido a que este dataset contaba con años desde el 2018 hasta el 2022 decidimos enfocarnos en el 2021, principalmente para analizar cómo influyó el COVID 19 en la industria de viajes

Data Explorer
Version 4 (25.78 GB)

- raw
- Airlines.csv**
- Combined_Flights_2018.c
- Combined_Flights_2018.p
- Combined_Flights_2019.c
- Combined_Flights_2019.p
- Combined_Flights_2020.c
- Combined_Flights_2020.p
- Combined_Flights_2021.c
- Combined_Flights_2021.p
- Combined_Flights_2022.c
- Combined_Flights_2022.p
- readme.html
- readme.md

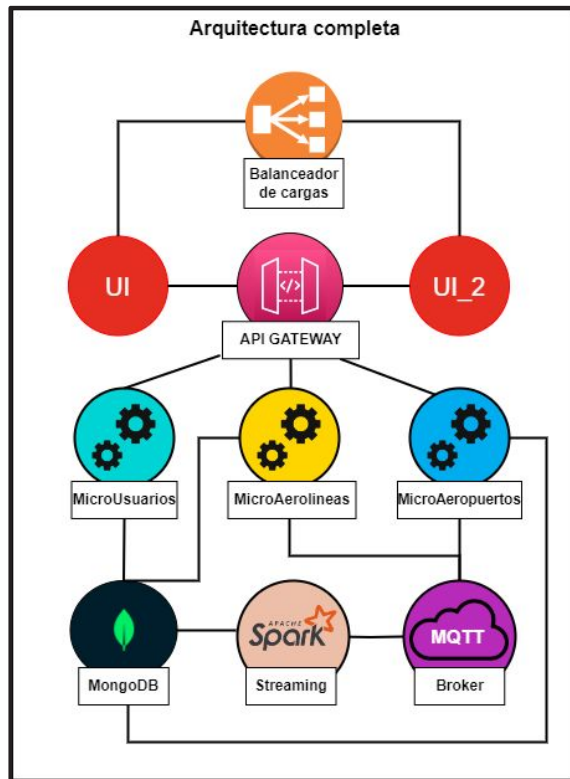
Combined_Flights_2021.csv (2.21 GB)
Detail Compact Column 10 of 61 columns

About this file
Combined flight data for entire year.

FlightDate	Airline	Origin	Dest	Cancelled
	Southwest Airlines... 17%	ATL 5%	ATL 5%	
2021-03-03	SkyWest Airlines Inc.	SGU	PHX	False
2021-03-03	SkyWest Airlines Inc.	PHX	SGU	False
2021-03-03	SkyWest Airlines Inc.	MHT	ORD	False



BBS71



Arquitectura completa del sistema

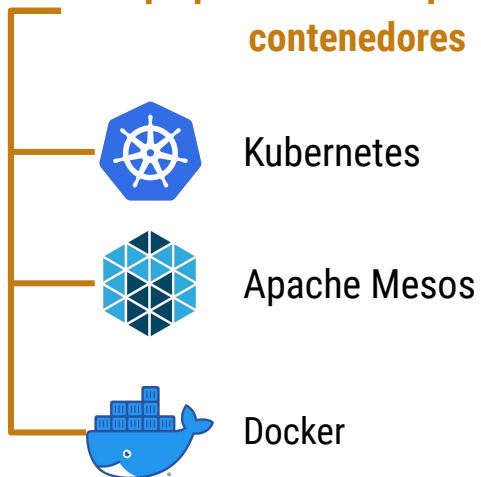
- Balanceador de cargas
- Api Gateway
- MicroUsuarios
- MicroAerolineas
- MicroAeropuertos
- MongoDB(Base de datos)
- Apache Spark
- Broker



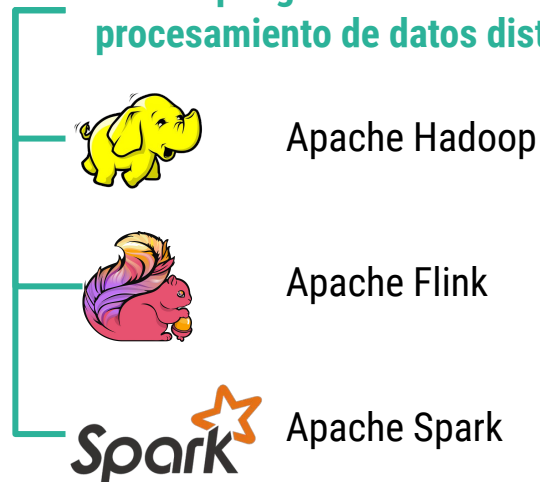


Generación y selección de alternativas de solución

Empaquetado de la aplicación en contenedores



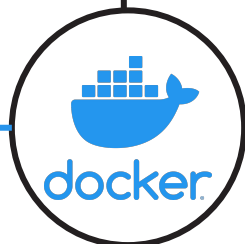
Despliegue en un cluster de procesamiento de datos distribuido





Alternativas seleccionadas

Para el empaquetado de la aplicación en contenedores, por los conocimientos que disponíamos de ella, por su escalabilidad y eficiencia



Para el despliegue en un cluster de procesamiento de datos distribuido por su compatibilidad e integración con otras aplicaciones.



BBS71



Arquitectura de la Pipeline

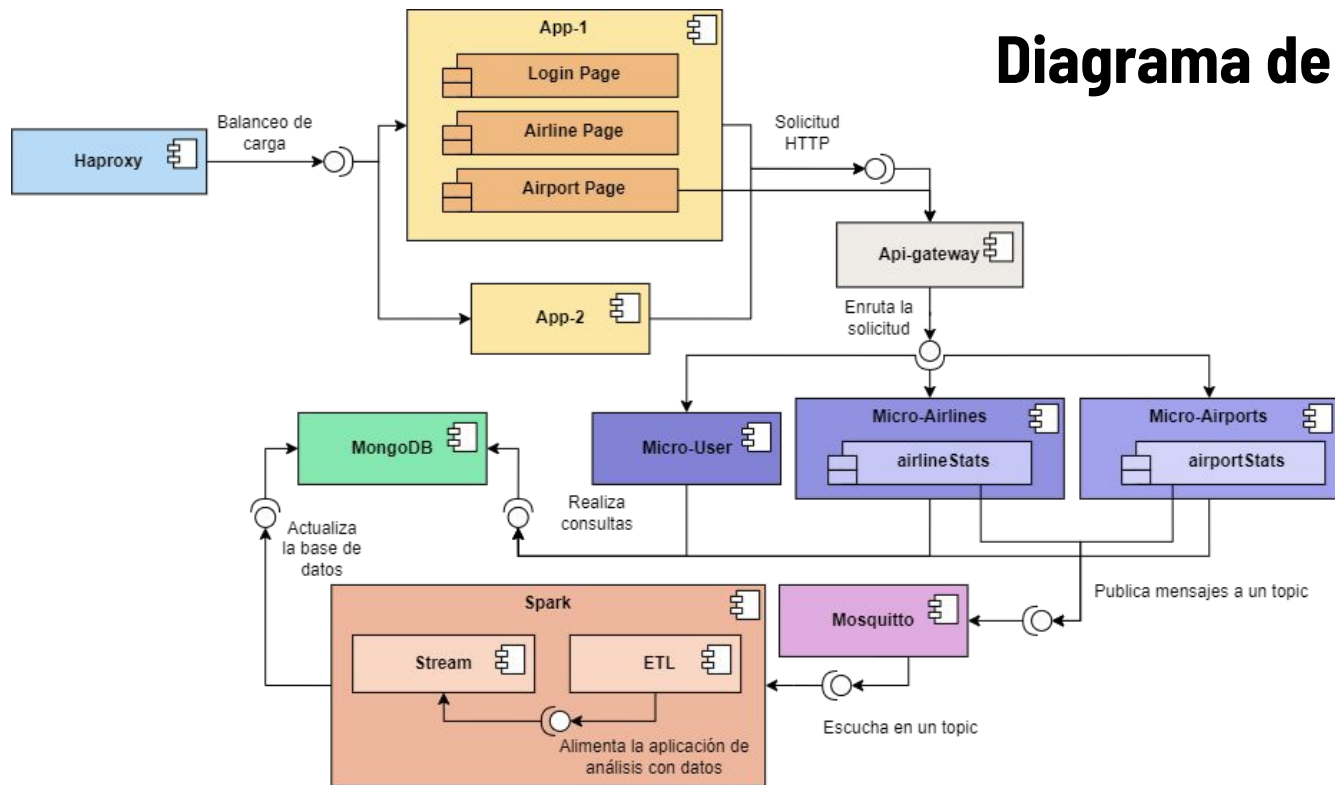




BBS71



Diagrama de componentes

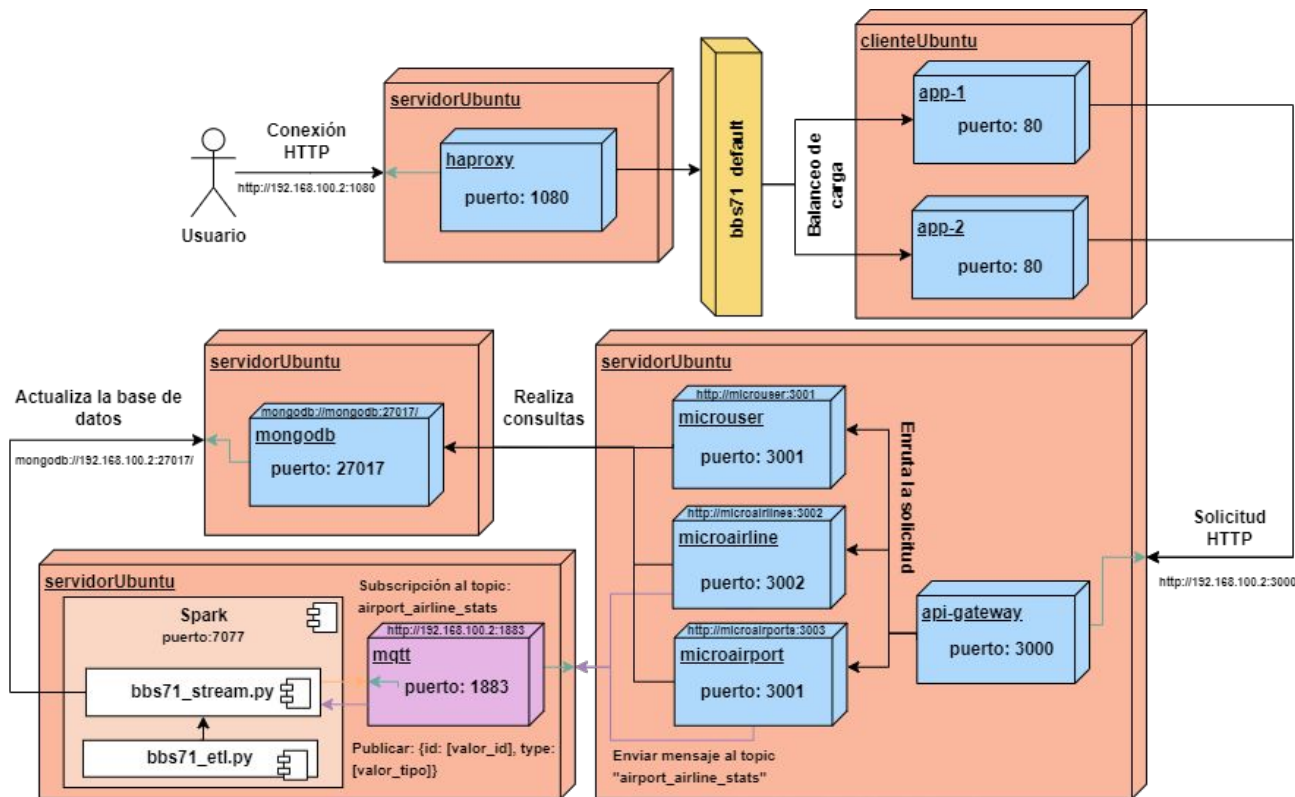




BBS71



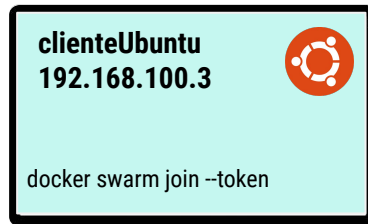
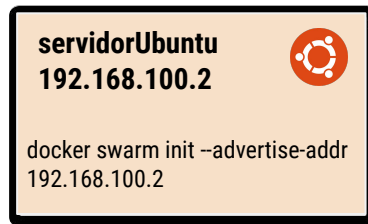
Diagrama de despliegue





Implementación de la solución diseñada

- 1) Se inicia un cluster de Docker Swarm con un nodo corriendo en el “**servidorUbuntu**” y otro en el “**clienteUbuntu**”.
- 2) Se hace un stack deploy para levantar los servicios que están dentro del Docker Compose yml.
- 3) Dentro del nodo clienteUbuntu se ejecutan únicamente los servicios de **app-1** y **app-2** que hacen referencia a la aplicación web.
- 4) En el nodo servidorUbuntu se corren los demás servicios; **mongodb**, **api-gateway**, **microuser**, **microairlines**, **microairports**, **haproxy**, **mqtt**





Implementación del análisis distribuido

- 5) Se debe tener iniciado un clúster de spark (master) con la dirección **spark://192.168.100.2:7077**
- 6) Lanzar un worker (slave) en la dirección del master, el cual se encarga de ejecutar los procesos de carga de trabajo
- 7) Se ejecutan las aplicaciones, primero **bbs71_etl.py** para generar los csv que utilizará la segunda aplicación

Bash

```
spark-submit --master spark://192.168.100.2:7077  
/home/vagrant/bbs71_git/bbs71_docker/spark_app/bbs71_stream.py
```





BBS71

Pruebas de escalabilidad y desempeño



Pruebas de carga normal

100 usuarios

Durante 30 sg. - 3 loops

Pruebas de carga alta

500 usuarios

Durante 30 sg. - 5 loops

Pruebas de estrés

1000 usuarios

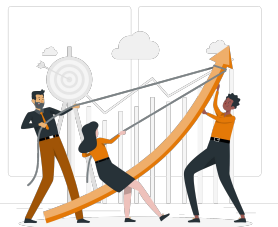
Durante 30 sg. - 10 loops

Api Gateway

Durante las pruebas se decidió escalar el servicio a dos réplicas para suplir un mejor desempeño. Los resultados gracias a esto mejoraron demostrando mejoras significativas.

Haproxy Balanceador de carga

Se cumplieron de forma satisfactoria las pruebas, demostrando resultados eficientes para el balanceador de cargas.



Aplicación de Streaming

Se identificó que la estructura actual del sistema presentaba un posible **cuello de botella**.



Se implementó una solución utilizando el enfoque de procesamiento paralelo. Gracias a esto **se optimizó el tiempo de respuesta y se redujo el impacto del cuello de botella**

Métricas a recolectar

- 1) Tiempo de respuesta promedio de las acciones realizadas por los usuarios.
- 2) Desviación estándar del tiempo de respuesta.
- 3) Porcentaje de error en las acciones realizadas.



BBS71



iGracias por la atención!

CREDITS: This presentation template was created by Slidesgo, and includes icons by Flaticon and infographics & images by Freepik

BOARDING PASS

- FLIGHT

B345

- GATE

D8

- SEAT

29E



Please keep this slide
for attribution



Referencias Bibliográficas

- [1] (2022) kaggle Website.[Online].Disponible en:
<https://www.kaggle.com/datasets/robikscube/flight-delay-dataset-20182022?select=Airlines.csv>
- [2] (2006) AWS amazon Kubernetes.[Online] Disponible en: <https://aws.amazon.com/es/kubernetes/>
- [3] (2012) Apache Mesos [Online]. Disponible en: <https://mesos.apache.org/>
- [4] (2013) Docker Website. [Online]. Disponible en:
<https://www.docker.com/products/container-runtime/>
- [5] (2014) IBM Website. [Online]. Disponible en: <https://www.ibm.com/mx-es/topics/apache-spark>
- [6] (2013) AWS amazon Hadoop. [Online]. Disponible en:
<https://aws.amazon.com/es/elasticmapreduce/details/hadoop/>
- [7] (2023) Aprender BIG DATA Apache Flink.[Online] Disponible en:
<https://aprenderbigdata.com/introduccion-apache-flink/>