

Telcom Customer Churn Prediction - MLOps Project

By Jorge Abrego Arevalo



Overview

The goal of this project is to address customer churn by building and deploying a machine learning model that predicts which customers are likely to leave the telco company. By leveraging the Telco customer churn dataset, which includes features such as customer demographics, location, services, monthly charges, and churn history, we can train a predictive model to proactively identify at-risk customers. This can help the business retain valuable customers by implementing targeted retention strategies.

To support the model's deployment and lifecycle management, the project will establish an MLOps infrastructure that focuses on the following key components:

- **ML Tracking:** Implement MLflow to record experiments, track hyperparameters, model versions, and performance metrics. This ensures reproducibility and keeps a detailed history of model iterations.
- **ML Deployment:** Use Docker as the primary deployment platform to deploy the churn prediction model into production environments, ensuring scalability and smooth delivery of predictions to business applications.
- **Workflows:** Automate workflows using Airflow to orchestrate the prediction and monitoring processes.
- **Model Monitoring:** Continuously monitor using Grafana to monitor the performance of the deployed model to detect issues like model drift, and ensure that predictions remain accurate over time.
- **Log Monitoring:** Implement centralized logging and log monitoring using ELK (Elasticsearch, Logstash, Kibana) to track logs for both the model and workflows. This ensures visibility into any issues, errors, or anomalies that may arise during pipeline execution or model performance.

This MLOps infrastructure will provide a robust foundation for operationalizing the churn prediction model and maintaining it throughout its lifecycle.

Dataset Information

The Telco Customer Churn dataset provides detailed information about a fictional telecommunications company's customers, aiming to predict which customers are likely to leave the service. It includes features such as demographic data (gender, dependents), location, monthly charges, subscribed services (e.g., internet, TV), and churn history. This dataset helps identify patterns in customer behavior that are related to churn, making it valuable for building predictive models to guide retention strategies. Source: [IBM Community](#).

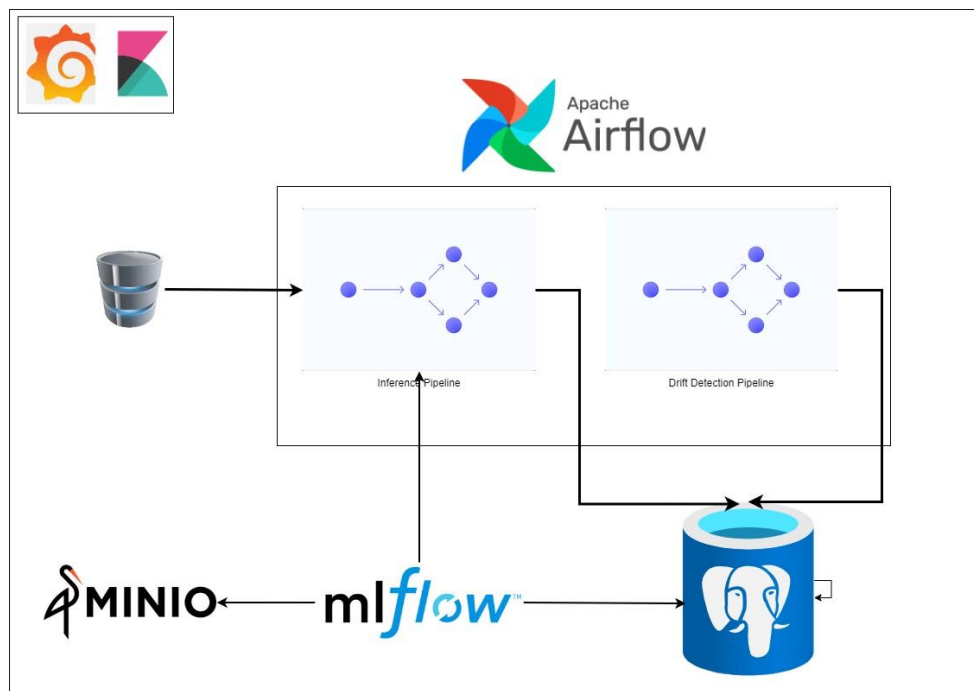
Inference

The Telco Customer Churn model is deployed using a pipeline in Airflow. The pipeline includes the following steps:

- Get data from database: Retrieve data from customer's database.
- Preprocess data: Clean and preprocess the data.
- Retrieve model: Retrieve the trained model from MLflow's model registry.
- Inference: Make predictions using the deployed model.
- Save data: Save the predictions to database, in order to be used for future analysis.

Solution Architecture

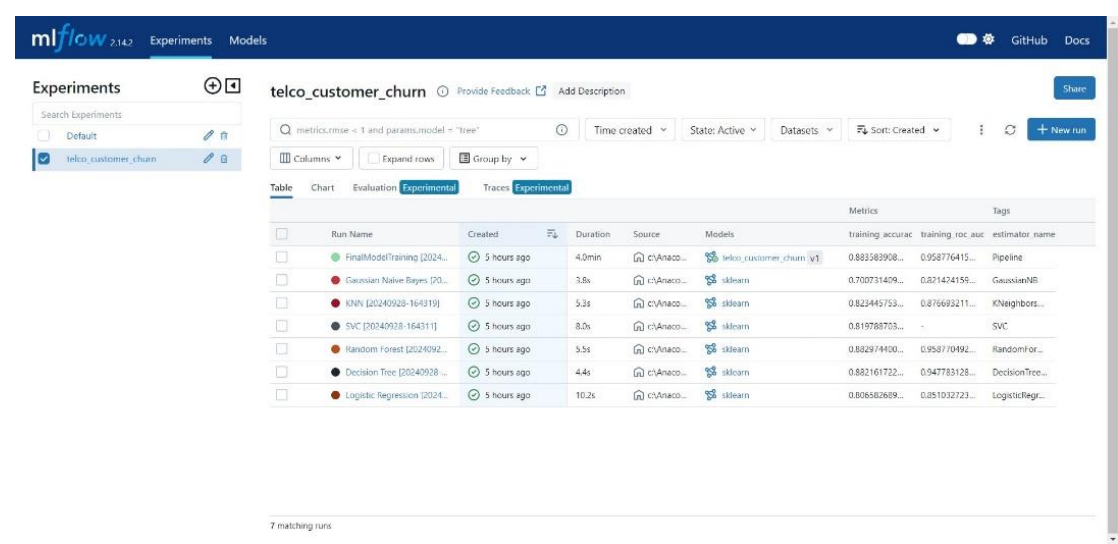
The following diagram illustrates the solution architecture for the Telco Customer Churn model.



Training Infrastructure

Consist of three main elements: MLflow, Postgres, and Minio.

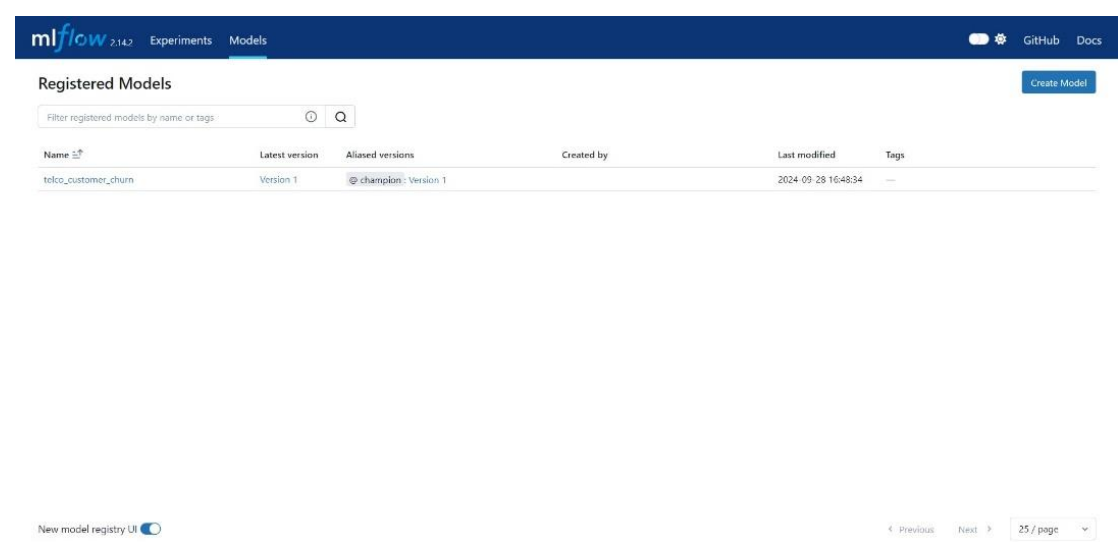
The MLflow component is responsible for tracking experiments, versioning models, and serving the final model.



The screenshot displays the MLflow Experiments page for the 'telco_customer_churn' experiment. The interface includes a sidebar with the experiment name, a search bar, and a 'New run' button. The main area shows a table of runs with columns for Run Name, Created, Duration, Source, Models, Metrics, and Tags. The table lists several runs, including 'FinalModelTraining [2024-09-28 16:43:11]', 'Gaussian Naive Bayes [2024-09-28 16:43:19]', 'KNN [2024-09-28 16:43:19]', 'SVC [2024-09-28 16:43:11]', 'Random Forest [2024-09-28 16:43:11]', 'Decision Tree [2024-09-28 16:43:11]', and 'Logistic Regression [2024-09-28 16:43:11]'. The 'Metrics' column shows training accuracy and ROC AUC values. The 'Tags' column lists the estimator names.

Run Name	Created	Duration	Source	Models	Metrics	Tags
FinalModelTraining [2024-09-28 16:43:11]	5 hours ago	4.0min	chAnaco...	telco_customer_churn v1	0.883589908... 0.958770415...	Pipeline
Gaussian Naive Bayes [2024-09-28 16:43:19]	5 hours ago	3.8s	chAnaco...	sklearn	0.700731409... 0.821424159...	GaussianNB
KNN [2024-09-28 16:43:19]	5 hours ago	5.3s	chAnaco...	sklearn	0.823445753... 0.876693211...	KNeighbors...
SVC [2024-09-28 16:43:11]	5 hours ago	8.0s	chAnaco...	sklearn	0.819788703... -	SVC
Random Forest [2024-09-28 16:43:11]	5 hours ago	5.5s	chAnaco...	sklearn	0.882974400... 0.958770492...	RandomFor...
Decision Tree [2024-09-28 16:43:11]	5 hours ago	4.4s	chAnaco...	sklearn	0.882161722... 0.947783128...	DecisionTree...
Logistic Regression [2024-09-28 16:43:11]	5 hours ago	10.2s	chAnaco...	sklearn	0.808582689... 0.851032723...	LogisticRegr...

MLFlow Experiments

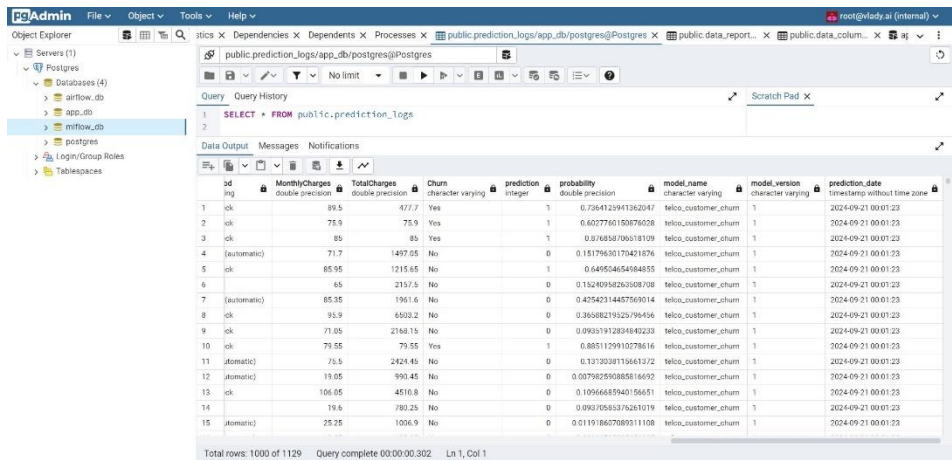


The screenshot displays the MLflow Models page. The interface includes a sidebar with the 'Models' tab selected, a search bar, and a 'Create Model' button. The main area shows a table of registered models with columns for Name, Latest version, Aliased versions, Created by, Last modified, and Tags. The table lists one model, 'telco_customer_churn', with version 'Version 1' and an alias 'champion: Version 1'. The 'Last modified' column shows the date '2024-09-28 16:48:34'.

Name	Latest version	Aliased versions	Created by	Last modified	Tags
telco_customer_churn	Version 1	champion: Version 1		2024-09-28 16:48:34	---

MLFlow Models

The Postgres component is used to store the data.

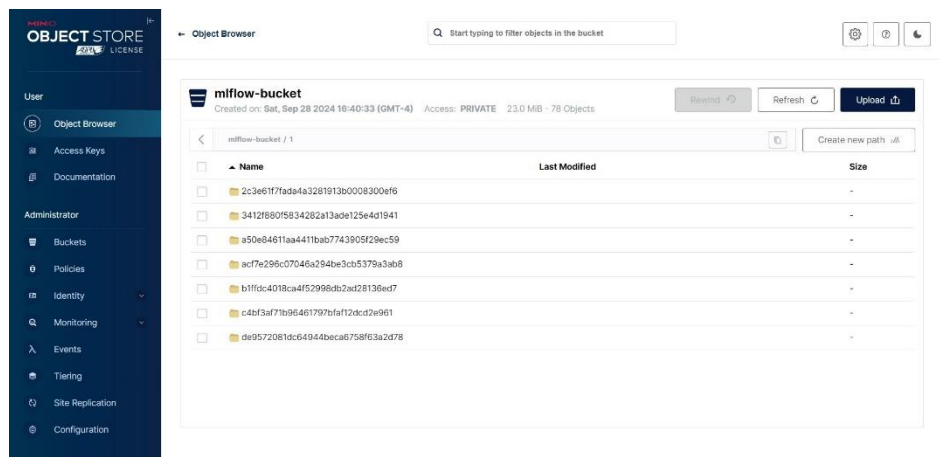


The screenshot shows the PgAdmin interface with a query executed: `SELECT * FROM public.prediction_logs`. The results are displayed in a table with 15 rows and 11 columns. The columns are: id, MonthlyCharges, TotalCharges, Churn, prediction, probability, model_name, model_version, and prediction_date. The data shows various customer records with their charges, churn status, and model predictions.

id	MonthlyCharges	TotalCharges	Churn	prediction	probability	model_name	model_version	prediction_date
1	89.5	477.7	Yes	1	0.7364125941362047	telco_customer_churn	1	2024-09-21 00:01:23
2	75.9	75.9	Yes	1	0.6027760150876028	telco_customer_churn	1	2024-09-21 00:01:23
3	85	85	Yes	1	0.876858706518109	telco_customer_churn	1	2024-09-21 00:01:23
4	71.7	1497.05	No	0	0.15179630170421876	telco_customer_churn	1	2024-09-21 00:01:23
5	85.95	1215.65	No	1	0.649504654684855	telco_customer_churn	1	2024-09-21 00:01:23
6	65	2157.5	No	0	0.15240916263508708	telco_customer_churn	1	2024-09-21 00:01:23
7	85.35	1961.6	No	0	0.42542314487569014	telco_customer_churn	1	2024-09-21 00:01:23
8	95.9	6503.2	No	0	0.36508219525795456	telco_customer_churn	1	2024-09-21 00:01:23
9	71.05	2168.15	No	0	0.09351912834840233	telco_customer_churn	1	2024-09-21 00:01:23
10	79.55	79.55	Yes	1	0.8851129910278616	telco_customer_churn	1	2024-09-21 00:01:23
11	75.5	2424.45	No	0	0.13130308115661372	telco_customer_churn	1	2024-09-21 00:01:23
12	19.05	990.45	No	0	0.007982590885816692	telco_customer_churn	1	2024-09-21 00:01:23
13	106.05	4510.8	No	0	0.10966685940156651	telco_customer_churn	1	2024-09-21 00:01:23
14	19.6	780.25	No	0	0.09370583376261019	telco_customer_churn	1	2024-09-21 00:01:23
15	25.25	1006.9	No	0	0.011918607089511108	telco_customer_churn	1	2024-09-21 00:01:23

PgAdmin client

The Minio component is used to store the model artifacts (including the final model, reference dataset, etc).



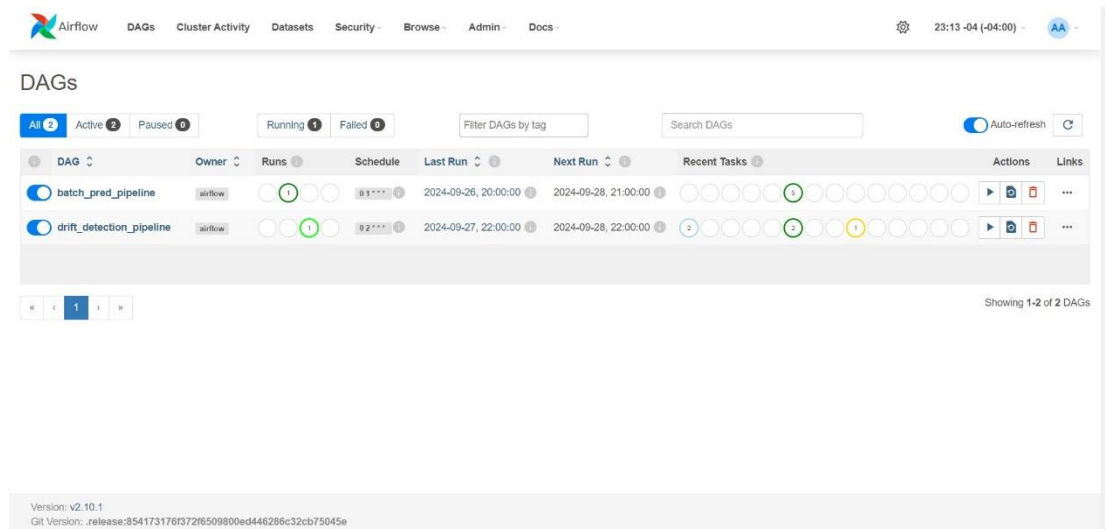
The screenshot shows the Minio Object Store interface. The left sidebar contains navigation links for User, Object Browser, Access Keys, Documentation, Administrator, Buckets, Policies, Identity, Monitoring, Events, Tiering, Site Replication, and Configuration. The main area displays the 'miflow-bucket' with a list of objects. The objects are listed with their names, last modified dates, and sizes.

Name	Last Modified	Size
2c3e61f7fada4a3281913b0008300ef6	-	-
3412f880f5934282a13ade125e4d1941	-	-
a50e84611aa4411bab7743905929ec59	-	-
ac7fe296c07046a284be3cb5379a3ab8	-	-
b1f1dc4018ca4f52986db2ad28136ed7	-	-
c4bf3af71b96461797bfaf12dcd2e961	-	-
de9572081dc64944beca6758f63a2d78	-	-

Minio Object Store

Workflow Infrastructure

Solution contains two pipelines: inference pipeline and monitoring pipeline. The inference pipeline is responsible for making predictions using the deployed model. The monitoring pipeline is responsible for monitoring the performance of the model. Each pipeline has a docker image containing the code for the pipeline.

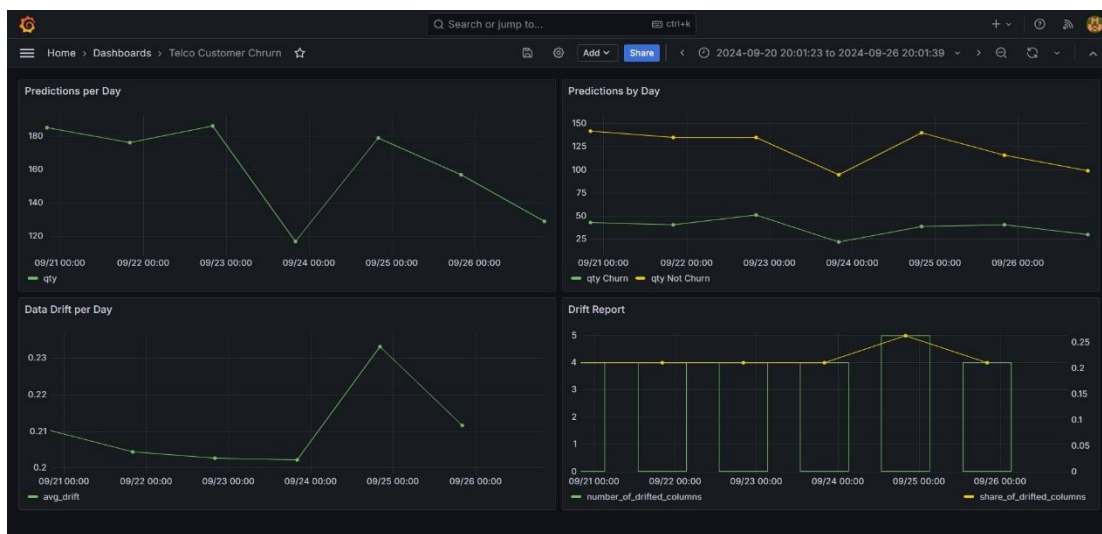


Airflow DAGs

Monitoring Infrastructure

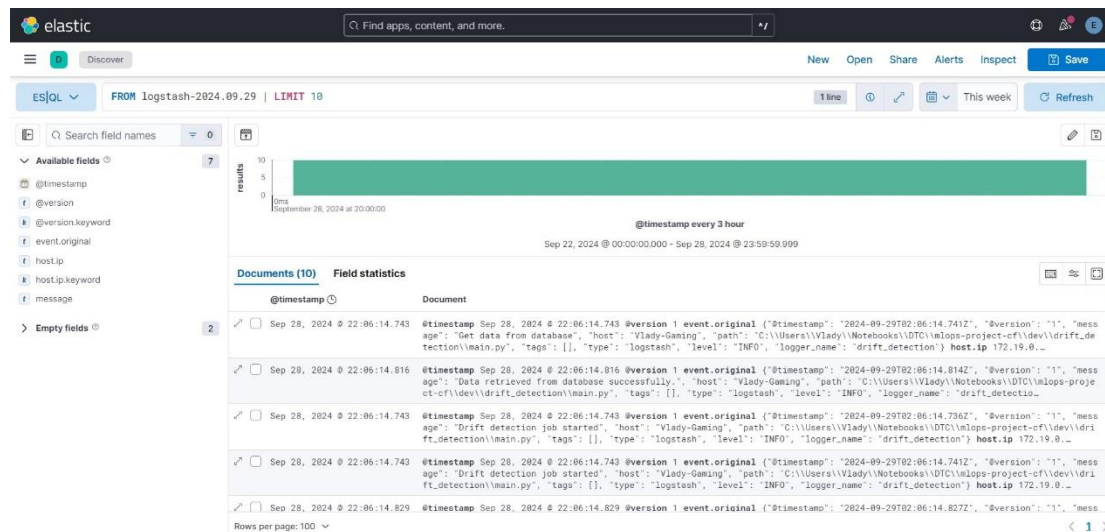
Consists in two elements: Grafana and ELK.

The Grafana component is used to monitor the performance of the deployed model.



Grafana Dashboard

The ELK component is used to monitor the logs of the model and workflows.



Kibana Monitor

Limitations

Below is a summary of the limitations of this solution:

Model Training

Modeling process depends on data scientists. ML Engineers should coordinate with Data Scientists on how to use MLflow to track the training process in their notebooks. Then, register the final model in registry to be used in production.

Inference and Monitoring Images Maintenance

ML Engineers should update the inference and monitoring images to the latest version when new features are added or code is modified.

Local Deployment Guide

The following guide provides detailed instructions on how to deploy the Telco Customer Churn model locally.

1. Clone the Repository:

```
> git clone https://github.com/JorgeAbrego/churn-prediction-mlops.git
```

```
> cd churn-prediction-mlops
```
2. Rename example.env to .env and replace the values with your own.

```
> move .env.example .env
```

```
> nano .env
```
3. Create a Python virtual environment and install the dependencies in requirements.txt:

```
> pip install -r requirements.txt
```

4. Build the Docker images required for the MLOps solution:

```
> docker build -t docker-socat:latest -f .\services\socat\Dockfile .\services\socat\  
  
> docker build -t mlflow-server:v2.12.2 -  
f .\services\mlflow\Dockfile .\services\mlflow\  
  
> docker build -t predict_batch:latest -f services/predict_batch/Dockfile  
services/predict_batch  
  
> docker build -t drift_detection:latest -f services/drift_detection/Dockfile  
services/drift_detection
```

5. Run the MLOps solution:

```
> docker-compose up -d
```

NOTE: Deploying the entire stack takes about 10 minutes while all checks and configurations are completed.

Once the MLOps solution is running, you can access to its different services using the following URLs:

- PGAdmin: <http://localhost:8888>
- MLflow: <http://localhost:5000>
- Airflow: <http://localhost:8080>
- Grafana: <http://localhost:3000>
- Kibana: <http://localhost:5601>
- Minio: <http://localhost:9001>