Qdox is a **command-line tool** executed locally within the activated Conda environment. It is meant to be run as a **standalone tool** for model diagnostics, not integrated into a live application.

**QDox requires a structured configuration file** that defines:

1. **Model Schema** (Features, Data Types, Expected Inputs)
2. **Training Data** (Path to the dataset used for model training)
3. **Reference Data** (Baseline data for comparison & drift detection)

QDox typically takes inputs in **YAML** or **JSON** format, which is passed using a configuration file.

Once the configuration file is set up, **the file is passed to QDox as a command-line argument**

Qdox generates an **HTML diagnostic report** summarizing findings.

**Step1 : Clone QDox and Set Up Conda (On an EC2 Instance)**

**Launch an EC2 instance** (Amazon Linux 2 or Ubuntu)

SSH into EC2 and Run the Following:

**Clone the QDox repository and set up the Conda environment**

# Clone QDox repository

git clone https://github.com/<your-org>/qdox.git

cd qdox

# Create and activate Conda environment

conda create --name qdox\_env python=3.8 -y

conda activate qdox\_env

**Install required dependencies.**

# Install dependencies

pip install -r requirements.txt

**Step 2: Set Up QDox Execution Script**

**Create a shell script to run QDox inside the EC2 instance.Inside your EC2 instance, create a shell script**

#!/bin/bash

source ~/anaconda3/bin/activate qdox\_env # Activate Conda

python3 /home/ec2-user/qdox/run\_qdox.py # Run QDox

Ensure the script is executable and correctly points to the QDox run command.

chmod +x /home/ec2-user/qdox/run\_qdox.sh

**Step 3 : Set Up QDox Configuration Files**

 **Create a YAML configuration file to specify:**

* Model schema (features, input types)
* Training dataset location
* Reference dataset location

 **This file needs to be updated dynamically with new data.**

**Step 4 : Enable AI/ML Model to Log Data in Splunk**

Ensure the ai/ml model logs predictions, input features, and confidence scores to Splunk.

The AI model must send logs to Splunk every time it makes a prediction.

QDox will use this logged data to analyze model drift and performance.

**Step 5 : Automate Data Extraction from Splunk (Every Hour)**

Use Splunk API or queries to fetch AI model logs from the last **one hour.**

This script fetches AI model logs every hour and writes them to a CSV file.

import requests

import json

import pandas as pd

# Splunk API details

splunk\_url = "https://splunk-server:8089/services/search/jobs/export"

splunk\_query = "search index=ai\_logs earliest=-1h latest=now | table timestamp, model\_name, input\_features, prediction, confidence\_score"

auth = ("admin", "your\_splunk\_password")

# Fetch logs

response = requests.post(splunk\_url, auth=auth, data={"search": splunk\_query})

# Convert to DataFrame and save

logs = json.loads(response.text)

df = pd.DataFrame(logs)

df.to\_csv(f"/data/datasets/live\_model\_data\_{pd.Timestamp.now().strftime('%Y-%m-%d\_%H-%M')}.csv", index=False)

print("Extracted Splunk logs and saved to CSV.")

**Step 6 :Automate QDox Config File Updates.**

Modify qdox\_config.yaml dynamically to point to the latest extracted data.

import yaml

import datetime

# Load existing config

config\_file = "/data/configs/qdox\_config.yaml"

with open(config\_file, "r") as file:

config = yaml.safe\_load(file)

# Update with latest dataset

latest\_data\_path = f"/data/datasets/live\_model\_data\_{datetime.datetime.now().strftime('%Y-%m-%d\_%H-%M')}.csv"

config["data"]["training\_data"] = latest\_data\_path

# Save updated config

with open(config\_file, "w") as file:

yaml.dump(config, file)

print(f"Updated QDox config file: {latest\_data\_path}")

Every hour, this script updates training\_data in the QDox config file.

**Step 7: Set Up AWS Lambda to Trigger QDox on EC2.**

AWS Lambda will be triggered whenever the config file is updated

import subprocess

def lambda\_handler(event, context):

# Step 1: Activate Conda environment

subprocess.run(["conda", "activate", "/data/ai/lib/02/env/ai/21/stable/prod/venv/conda\_envs/qdox\_prod"])

# Step 2: Run QDox

subprocess.run(["qdox", "--config", "/data/configs/qdox\_config.yaml"])

# Step 3: Upload results to S3

subprocess.run(["aws", "s3", "cp", "/data/reports/qdox\_outputs", "s3://my-bucket/qdox-reports/", "--recursive"])

return {"status": "QDox Execution Completed"}

Schedule QDox to Run Every Hour

Store QDox Reports in Amazon S3

AWS Lambda Triggers : Set CloudWatch to trigger every hour when the QDox config file is modified.

Lambda will:

Activate Conda in EC2

Run QDox with the updated config

Upload QDox reports to S3 for monitoring

**Step 8: QDox Runs and Generates Reports**

* QDox processes the latest training and reference data.
* Produces an HTML or Confluence-based report summarizing:
  + Drift detection
  + Fairness analysis
  + Model diagnostics

**Step 9: Store QDox Reports in Amazon S3**

* Once the QDox report is generated, upload it to an S3 bucket for further monitoring and access.

**. Business & IT Requirements**

* **Business Requirements** for MAC have been finalized and signed off by Danny (*confirmation from Seraphina pending*).
* **IT Requirements** have been drafted, covering AI observability needs.
* Confirmed with EA that **LLMs will only be used offline**, eliminating the need for AI observability and associated approvals (excluding Auth & duplicate tracks).

**2. IT Requirements Breakdown**

* **Metric Requirements**
  + AI/ML metrics: Performance, drift, data integrity, and traffic metrics have been identified and documented.
  + Logging requirements: Information/parameters for tracking have been documented.
  + LLM metrics have been identified and documented.
* **Data & Pipeline Requirements**
  + Currently in progress.
  + These requirements will be refined based on the AI observability tool.
* **Infrastructure & Production Support Requirements**
  + Identified and currently under review by Tom.

**3. RAI (Responsible AI) Requirements**

* Review by RAI team on the list of AI/ML & LLM metrics is in progress.
* **ETA: To be determined.**

**4. Sign-offs**

* **LL Track PO Sign-off**: In progress.
* **IT Requirements for MAC Block PO Sign-off**: Will be initiated.

**5. Qdox Assessment & Coverage Analysis**

* Conducted an analysis of AI/ML metrics covered by **Qdox** and identified gaps.
* For AI/ML metrics **not covered by Qdox**, a **minor build** is required.
* Awaiting confirmation of **ETA from the product team**.
* Requested additional details on **Qdox functionality and capabilities**.

**6. Next Steps on Qdox**

* **Demo/Walkthrough** requested.
* Documentation does not list **catboost** as supported—following up with the product team.
* LLM metrics are not covered by **Qdox**, but per **Bryce**, these could be built. Following up on ETA.

**7. Alternative Solutions - SageMaker Monitoring**

* Conducted a **comparison of SageMaker metrics monitoring vs. Qdox**—results published in Confluence.
* Additional research on **cost comparison and feasibility** is required.
* Preliminary **cost estimate for SageMaker model monitoring** is available; a more detailed estimate is **in progress**.
* **Seraphina to provide ETA** for the detailed cost estimate.

**8. Next Steps & Action Items**

* **Assessment of Qdox and alternatives**: Target completion by **March 11**.
* Evaluate **SageMaker monitoring tool** in parallel.