Cairo Traffic Dashboard (Azure Event Hub Only) – Documentation

This document explains the architecture, components, flow, and code structure of the **Azure-Connected Cairo Traffic Dashboard** based on the last code you received.

---

1. Overview

The Cairo Traffic Dashboard is a **real-time monitoring system** that receives traffic events from **Azure Event Hub** and displays them in Streamlit through:

- Live Overview
- Map Visualization (PyDeck)
- Analytics (Plotly)
- Alerts & Anomalies (Toast Notifications)

No local simulator is used – **Azure Event Hub is the only data source**.

---

2. Architecture Diagram

```
Azure Event Hub  --->  Streamlit App  --->  UI (Overview / Map / Analytics /
Alerts)
                           |
                           -->  Alert Engine
```

---

3. Data Flow

1. Event Hub pushes JSON messages.
2. Streamlit client receives messages using:
3. `EventHubConsumerClient.receive()`
4. Each message is parsed and stored in:
5. `st.session_state.df` (all events)
6. Anomaly rules check each event:
7. Low speed
8. High speed
9. High congestion
10. Over capacity
11. Traffic incidents
12. Alerts added to:
13. `st.session_state.alerts`
14. Real-time toast notifications appear.

---

4. Components Breakdown

4.1. Streamlit Sidebar

Contains: - Start Listening - Stop Listening - Tab visibility toggles

4.2. Event Hub Consumer

Creates persistent listener:

```
client = EventHubConsumerClient.from_connection_string(...)
client.receive(on_event=on_event)
```

4.3. Event Handler (`on_event`)

Executed per message: - Parse JSON - Append to dataframe - Detect anomalies - Add alerts - Show toast notifications

4.4. Tabs

Overview Tab

Shows last 50 events + latest event JSON.

Map Tab

Uses PyDeck ScatterplotLayer to show congestion visualization.

Analytics Tab

Uses Plotly to graph: - Vehicle count over time - Average speed over time - Vehicle type distribution

Alerts Tab

Displays alert table + bar chart of alert types.

---

5. Anomaly Rules

```
Speed < 10 km/h         → Low speed
Speed > 100 km/h        → High speed
Congestion > 120%       → Over capacity
Congestion > 85%        → High congestion
TrafficIncident != None → Incident alert
```

---

6. Core Functions

6.1. detect_anomaly(event)

Evaluates event data and returns list of alert messages.

6.2. on_event(partition_context, event)

Main logic for: - Parsing data - Storing events - Detecting anomalies - Triggering popups

---

7. Azure Requirements

- Event Hub Namespace
- Event Hub name
- SAS Key (RootManageSharedAccessKey)
- Connection String

App uses:

```
consumer_group = "$Default"
starting_position = "-1"  # Read from beginning
```

---

8. File Structure

```
/your-app
 ├── app.py (Streamlit dashboard)
 ├── requirements.txt
 └── README.md (this documentation)
```

---

9. How to Run

Install dependencies:

```
pip install streamlit azure-eventhub pydeck plotly pandas
```

Run:

```
streamlit run app.py
```

Click **Start Listening** in the sidebar.

---

10. Error Handling

Common issues:

- • Wrong Event Hub name
- • Wrong connection string
- • Firewall blocking traffic
- • No messages being sent

---

11. Expansion Options

You can later add: - Azure Blob storage archiving - Azure SQL analytics - Real map tiles - Incident heatmap layers - Filtering per-location

---

12. Appendix: Full Code Reference

The full Azure-only dashboard code is included exactly as delivered in the last message, with all components preserved (tabs, map, charts, alerts, etc.).

(Place full code below here if needed.)

End of Documentation