📌 Replacing Telegram with PostgreSQL in G.E.N.I.E.'s Workflow

✅ **Final Decision: PostgreSQL ONLY → No Slack or Airtable for now.**
✅ **Goal:** PostgreSQL will replace Telegram **for trade logging, execution tracking, and risk monitoring.**
✅ **Next Step:** Modify **n8n workflow** to send **trade alerts & execution data** directly to PostgreSQL instead of Telegram.

🛠 Step-by-Step Guide: Replace Telegram with PostgreSQL in n8n

Here's how to completely remove Telegram and replace it with **PostgreSQL** for storing trade alerts, execution logs, and institutional flow tracking.

Step 1: Create a PostgreSQL Table for Trade Logging

First, we **set up a table** in PostgreSQL to store all execution alerts.

1️⃣ Connect to PostgreSQL

Run the following command to enter the PostgreSQL database:

```
psql -U admin -d trading_db
```

2️⃣ Create a New Table for Trade Execution Logs

Run this SQL query to create a table:

```
CREATE TABLE trade_logs (
    trade_id SERIAL PRIMARY KEY,
    timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
```

```
    symbol VARCHAR(10),
    trade_type VARCHAR(10),
    price DECIMAL(10,2),
    smartflow_score DECIMAL(5,2),
    execution_status VARCHAR(20)
);
```

✅ **This table will store:**

- **Trade ID** (auto-generated)
- **Timestamp** (when the trade was logged)
- **Symbol** (SPY, ES, NQ, etc.)
- **Trade Type** (BUY or SELL)
- **Price** (Execution price)
- **SmartFlow Score** (Institutional sentiment tracking)
- **Execution Status** (e.g., "FILLED", "FAILED", "CANCELLED")

Step 2: Modify n8n Workflow to Use PostgreSQL Instead of Telegram

Now, we replace the **Telegram node** in **n8n** with a **PostgreSQL node**.

1️⃣ Open n8n & Locate the Telegram Node

- Go to **n8n Dashboard**
- Open the **workflow where Telegram is used**
- Find the **Telegram Node** (Usually named **"Send Trade Alert"**)

2️⃣ Delete the Telegram Node

- Click on the **Telegram Node**
- Press **Delete** to remove it from the workflow

3️⃣ Add a PostgreSQL Node

- Click **Add Node**
- Search for **PostgreSQL**
- Select **“Insert”** (for adding new trade logs)

4️⃣ Configure the PostgreSQL Node

- **Database Credentials:**
  - Host: `localhost` (or your remote PostgreSQL server)
  - Database: `trading_db`
  - User: `admin`

    - Password: `your_password`
- **SQL Query for Trade Logging:**
- `INSERT INTO trade_logs (symbol, trade_type, price, smartflow_score, execution_status)`
- `VALUES ('{{ $json.symbol }}', '{{ $json.trade_type }}', {{ $json.price }}, {{ $json.smartflow_score }}, 'FILLED');`

✅ **This replaces Telegram by logging trades directly into PostgreSQL.**

Step 3: Test the New PostgreSQL Workflow

1️⃣ Run a Test Trade Execution

- In **n8n**, manually execute a test trade.
- Check if the trade **logs into PostgreSQL**.

2️⃣ Verify Trade Logs in PostgreSQL

Run this SQL command to check if the trades are being stored correctly:

```
SELECT * FROM trade_logs ORDER BY timestamp DESC LIMIT 5;
```

✅ If you see recent trades **logged into the database**, the Telegram replacement is successful.

Step 4: Automate & Optimize Trade Logging

Once PostgreSQL is integrated into **n8n**, ensure the system **automatically logs every trade**.

1️⃣ Enable Auto-Logging in n8n

- Click the **PostgreSQL Node**
- Set **"Execute Automatically"** for every trade execution
- Save & Activate Workflow

2️⃣ Confirm Execution Logging

Run a few trades and verify the logs:

```
SELECT * FROM trade_logs ORDER BY timestamp DESC;
```

✅ **This confirms all trades are logged in real-time.**

🚀 Final Check – Telegram Fully Removed, PostgreSQL Replacing It

Feature	Old (Telegram)	New (PostgreSQL)
Trade Alerts	Sent via chat	Stored in DB
Execution Logging	No storage	Saved in PostgreSQL
Risk Monitoring	No tracking	Logged for analysis
Scalability	API limits	Institutional-grade

🔥 Next Steps

✅ **PostgreSQL is now replacing Telegram for execution logs & tracking.**
✅ **Every trade will be stored for analysis, risk monitoring, and AI execution.**
✅ **We can now expand to include risk alerts & real-time dashboards.**

🚀 Do you need help setting up queries for risk tracking & real-time dashboards in PostgreSQL? Let me know! 🔥