**Project Guide: MQTT-to-PostgreSQL Pipeline**

*Automatically store ESP32 sensor data in a PostgreSQL time-series database*

**Phase 1: Modular Testing**

**Component A: MQTT → Python**

**Objective**: Validate data reception from Mosquitto.  
**Steps**:

1. Write a Python script using paho-mqtt to:
   * Connect to localhost:1883.
   * Subscribe to a test topic (e.g., esp32/test).
   * Print incoming messages to the console.
2. Test with the mosquitto\_pub CLI:

bash

Copy

mosquitto\_pub -t "esp32/test" -m "test\_message"

1. Ensure the script runs continuously (use loop\_forever()).

**Component B: Python → PostgreSQL**

**Objective**: Validate database writes.  
**Steps**:

1. Design a time-series table:

sql

Copy

CREATE TABLE sensor\_data (

time TIMESTAMPTZ DEFAULT NOW(),

device\_id TEXT,

value FLOAT

);

1. Write a Python script using psycopg2 to:
   * Connect to PostgreSQL (localhost:5433).
   * Insert hardcoded test data.
   * Query the table via pgAdmin to confirm.

**Phase 2: Data Integration**

**Objective**: Process MQTT data for database insertion.  
**Steps**:

1. Modify the MQTT script to:
   * Parse payloads (e.g., JSON to Python dict).
   * Add logging for parsed data.
2. **Do not connect to the database yet** – isolate parsing logic.

**Phase 3: End-to-End Workflow**

**Objective**: Connect all components.  
**Steps**:

1. Add database insert logic to the MQTT script.
2. Implement error handling:
   * Retry failed database connections.
   * Log insertion errors to a file.
3. Test with live ESP32 data.

**Phase 4: Automation**

**Objective**: Run the script as a background service.  
**Options**:

* **Linux**: Create a systemd service.
* **Windows**: Use Task Scheduler or nssm.

**Key Considerations**

1. **Security**:
   * Store credentials in environment variables.
   * Restrict PostgreSQL access to localhost.
2. **Performance**:
   * Batch inserts for high-frequency data.
   * Use PostgreSQL’s TIMESERIES extension (optional).
3. **Monitoring**:
   * Log script activity (e.g., logging module).
   * Set up alerts for failures.

**Debugging Tips**

1. Test MQTT with mosquitto\_sub -t "#" -v.
2. Use pgAdmin to manually verify table entries.
3. Add debug print statements between each processing step.

**Next Steps**

1. Start with **Component A**.
2. Design your database schema early.
3. Integrate only after both components work independently.

**To save as a PDF**:

1. Copy this text into a Word/Google Docs file.
2. Adjust formatting as needed.
3. Go to **File → Download → PDF**.

Let me know if you need help with specific sections! 🛠️