PLANT WILTING DETECTION SYSTEM

End-to-End IoT & Data Engineering Pipeline

Portfolio Project Technical Documentation

Version	1.0.0
Date	March 2026
Status	Production

IoT | Cloud | Data Engineering | Data Analysis | Business Intelligence

Mark John V. Merana
Junior Data Analyst | Computer Engineer

1. Project Overview

This document provides comprehensive technical documentation for the Plant Wilting Detection System — a full-stack IoT and data engineering solution designed to monitor greenhouse plant health in real time and surface actionable insights through an interactive Power BI dashboard.

The system continuously collects environmental sensor data (temperature, humidity, soil moisture) from three monitored plant specimens. Readings are transmitted wirelessly via ESP32 microcontrollers, ingested to Google Sheets through Apps Script, extracted via the Google Sheets API, orchestrated through Apache Airflow pipelines, transformed and persisted in a SQL Server data warehouse, and finally visualized in Power BI.

1.1 Business Objectives

- Detect early-stage plant wilting before visible symptoms appear
- Reduce plant mortality rates in controlled greenhouse environments
- Provide real-time and historical trend analysis for crop health
- Enable automated alerting when wilting risk scores exceed safe thresholds
- Build a reproducible, scalable IoT pipeline suitable for production deployment

1.2 Key Metrics Tracked

Metric	Description
Wilt Risk Score	Composite score (0.0–1.0) derived from ML confidence model
Wilted %	Percentage of readings where wilting was classified as positive
Humidity Difference	Delta between inside and outside relative humidity
Temp Difference	Delta between inside and outside temperature (°C)
AVG Inside Temp	Mean internal greenhouse temperature per day
AVG Moisture	Mean soil moisture reading across all plant sensors
Total Readings	Cumulative sensor readings ingested in the pipeline
Confidence Score	Per-plant ML model confidence for wilting detection (%)

2. System Architecture

The pipeline is structured as a seven-layer architecture following industry-standard separation of concerns: data acquisition, edge transmission, cloud ingestion, orchestration, transformation, storage, and visualization.

#	Layer	Technology	Responsibility
01	Sensor Layer	DHT22 / Capacitive Soil Moisture Sensor	Captures temperature, humidity, soil moisture
02	Edge Layer	ESP32 Microcontroller (Wi-Fi enabled)	Aggregates sensor data and transmits over HTTPS
03	Ingestion Layer	Google Sheets + Apps Script	Receives and logs raw telemetry in cloud spreadsheet
04	Extraction Layer	Google Sheets API v4	Programmatic read access for downstream ETL
05	Orchestration Layer	Apache Airflow (DAG-based)	Schedules, monitors, and retries ETL pipeline runs
06	Transformation Layer	Python (Pandas, Numpy)	Cleans, enriches, and manipulation on data
07	Storage Layer	Microsoft SQL Server	Hosts the normalized data warehouse / queries
08	Visualization Layer	Microsoft Power BI	Interactive dashboard with DAX measures and drill-through

Pipeline Flow: Physical Sensors → ESP32 → Google Sheets (Apps Script) → Google Sheets API → Apache Airflow → Python ETL → SQL Server → Power BI Dashboard

3. Layer 1 — Sensor Hardware

3.1 Components Used

Component	Specification
Temperature & Humidity Sensor	DHT22 — Range: -40°C to +80°C / 0–100% RH, ±0.5°C accuracy
Soil Moisture Sensor	Capacitive v1.2 — Analog 0–3.3V output, corrosion-resistant
Power Supply	3.3V regulated via ESP32 onboard regulator
Sampling Rate	One reading per plant per 60 seconds
Number of Plants Monitored	3 (Plant 1, Plant 2, Plant 3)
Sensor Placement	DHT22 mounted 3m above plant canopy; moisture probe inserted 5cm into soil

3.2 Sensor Readings Schema

Each polling cycle produces the following raw data payload per plant:

Field	Data Type & Units
plant_id	INT — Identifier 1, 2, or 3
inside_temperature	FLOAT — Degrees Celsius (°C)
inside_humidity	FLOAT — Relative Humidity (%)
outside_temperature	FLOAT — Degrees Celsius (°C)
outside_humidity	FLOAT — Relative Humidity (%)
soil_moisture	FLOAT — Percentage (0–100%)
timestamp	DATETIME — ISO 8601 UTC format

4. Layer 2 — ESP32 Edge Controller

4.1 Overview

The ESP32 (Espressif Systems) serves as the edge computing node. It polls the attached sensors, packages the readings into a JSON payload, and transmits them to Google Sheets via an HTTPS POST request to a deployed Apps Script Web App URL. The ESP32 uses its built-in Wi-Fi module (802.11 b/g/n) to establish a secure connection.

4.2 Firmware Logic

The firmware is written in C++ using the Arduino framework. Key responsibilities:

- Initialize I2C/GPIO interfaces for DHT22 and capacitive moisture sensor
- Establish Wi-Fi connection using stored SSID and password credentials
- Poll sensors every 60 seconds on a non-blocking timer (millis() based)
- Serialize readings to JSON and POST to Apps Script deployment URL
- Implement exponential backoff retry logic on HTTPS failure (max 3 retries)
- Deep sleep between reads to conserve power (optional low-power mode)

4.3 Sample JSON Payload

```
{
  "plant_id": 1,
  "inside_temp": 33.52,
  "inside_humidity": 65.40,
  "outside_temp": 32.48,
  "outside_humidity": 57.98,
  "soil_moisture": 72.56,
  "timestamp": "2026-03-06T08:42:00Z"
}
```

4.4 Wi-Fi & Security Considerations

Concern	Implementation
Credential Storage	Stored in ESP32 NVS (Non-Volatile Storage) flash partition
Transport Security	HTTPS with TLS 1.2 enforced; root CA certificate pinned in firmware
Endpoint Authentication	Apps Script URL acts as shared secret; guessable URLs are rotated monthly
Error Handling	3-attempt retry with 5s / 15s / 60s exponential backoff
Data Validation	Sensor plausibility checks: temp > -10°C, humidity > 0%, moisture > 0%

5. Layer 3 — Google Sheets & Apps Script

5.1 Architecture

Google Sheets serves as the primary cloud-based raw data store. A Google Apps Script Web App acts as an HTTP endpoint that receives POST requests from the ESP32 and appends rows to the designated spreadsheet. This approach is chosen for its zero-cost ingestion tier, ease of setup, and native integration with Google APIs.

5.2 Apps Script Web App

The Apps Script is deployed as a Web App with the following configuration:

Parameter	Value
Execute As	Me (service account owner)
Who Has Access	Anyone (anonymous POST supported)
Deployment Type	Production deployment with versioned releases
Trigger	HTTP POST to Web App URL
Sheet Target	Tab named 'Data' in master spreadsheet
Column Order	Timestamp | Plant_ID | Status | Confidence Score | Inside_Temp | Inside_Humidity | Outside_Temp | Outside_Humidity | Soil_Moisture

5.3 Apps Script Code Logic

```
function doPost(e) {
  const data = JSON.parse(e.postData.contents);
  const sheet = SpreadsheetApp
    .openById('SPREADSHEET_ID')
    .getSheetByName('SensorLog');
  sheet.appendRow([
    new Date(),
    data.plant_id,
    status,
    confidence score,
    data.inside_temp,
    data.inside_humidity,
    data.outside_temp,
    data.outside_humidity,
    data.soil_moisture
  ]);
  return ContentService
    .createTextOutput(JSON.stringify({ status: 'ok' }))
    .setMimeType(ContentService.MimeType.JSON);
}
```

5.4 Data Volume & Retention

Parameter	Value
Rows per Day	~4,320 rows/day (3 plants × 1 read/min × 1,440 min)
Total Readings (Current)	3,984 readings
Retention Policy	Raw logs retained for 90 days; archived to Google Drive as CSV monthly
Max Sheet Rows	10,000,000 cells (Google Sheets limit)

6. Layer 4 — Google Sheets API Extraction

6.1 Overview

The Google Sheets API v4 is used by the Python ETL process (orchestrated via Airflow) to extract raw sensor data for downstream processing. The API provides structured access to spreadsheet data as JSON, enabling reliable programmatic extraction without manual CSV exports.

6.2 Authentication

Property	Detail
Auth Method	Google Service Account with JSON key file
OAuth Scope	https://www.googleapis.com/auth/spreadsheets.readonly
Key Storage	Encrypted at rest; injected via Airflow Variable or Kubernetes Secret
Library Used	google-auth, google-api-python-client

6.3 Extraction Pattern

Data is extracted using an incremental load strategy — only rows newer than the last successfully processed timestamp are fetched. The high-water mark timestamp is stored in Airflow's XCom system.

```
from googleapiclient.discovery import build
from google.oauth2.service_account import Credentials

SCOPES = ['https://www.googleapis.com/auth/spreadsheets.readonly']
creds = Credentials.from_service_account_file('key.json', scopes=SCOPES)
service = build('sheets', 'v4', credentials=creds)

result = service.spreadsheets().values().get(
    spreadsheetId=SPREADSHEET_ID,
    range='SensorLog!A:G'
).execute()
rows = result.get('values', [])
```

7. Layer 5 — Apache Airflow Orchestration

7.1 Overview

Apache Airflow serves as the pipeline orchestrator. A Directed Acyclic Graph (DAG) defines the sequence of ETL tasks, handles scheduling, provides retry logic, and enables end-to-end monitoring and alerting for the data pipeline.

7.2 DAG Configuration

Property	Value
DAG ID	plant_wilting_etl_pipeline
Schedule Interval	Every 30 minutes (@every 30 min cron: */30 * * * *)
Start Date	2026-01-01
Catchup	False (no historical backfill on deploy)
Max Active Runs	1 (prevents concurrent pipeline conflicts)
Retry Policy	retries=3, retry_delay=timedelta(minutes=5)
Owner	Mark John Merana
Tags	iot, greenhouse, wilting, production

7.3 DAG Task Sequence

Task ID	Operator	Description
task_check_connection	PythonOperator	Verify the SQL Server connection before doing any work.
task_get_watermark	PythonOperator	Read the last processed timestamp and push to XCom.
task_extract	PythonOperator	Extract new rows from Google Sheets since the watermark.
task_transform	PythonOperator	Clean and validate the raw data.
task_load	SimpleHttpOperator	Upsert clean data into SQL Server.
task_update_watermark	PythonOperator	Advance the watermark to the latest timestamp in this batch.

7.4 Monitoring & Alerting

- Airflow UI available at localhost:8080 (or via reverse proxy in production)
- All task logs persisted to cloud storage (GCS bucket) for 30-day retention
- On-failure email alerts configured via SMTP connection in Airflow
- SLA miss alerts if total pipeline run exceeds 10 minutes

8. Layer 6 — Python ETL

8.1 Environment & Dependencies

Package	Purpose
pandas >= 2.0	DataFrame manipulation, cleaning, and transformations
numpy >= 1.24	Numerical computation and array operations
sqlalchemy >= 2.0	ORM-based SQL Server connection and upsert operations
pyodbc	ODBC driver for SQL Server connectivity
google-auth	Google Service Account authentication
google-api-python-client	Google Sheets API client
apache-airflow >= 2.7	Pipeline orchestration framework

8.2 Data Cleaning Steps

- Drop rows where all sensor readings are null (sensor disconnection events)
- Impute isolated null moisture readings with forward-fill (max 3 consecutive)
- Clip temperature values to physically valid range: -10°C to 60°C
- Clip humidity values to 0%–100%
- Clip moisture values to 0%–100%
- Remove duplicate rows with same (plant_id, timestamp) combination
- Convert timestamps to UTC and normalize to minute-level precision

8.3 Feature Engineering

Feature	Derivation Logic
humidity_diff	inside_humidity - outside_humidity
temp_diff	inside_temperature - outside_temperature

9. Layer 7 — SQL Server Data Warehouse

9.1 Schema Design

The data warehouse optimized for Power BI analytical queries. A central table stores all sensor readings.

9.2 Table Definitions

Fact Table: fact_sensor_readings

Column	Data Type & Constraints
reading_id	BIGINT IDENTITY(1,1) PRIMARY KEY
plant_key	INT FOREIGN KEY → dim_plant(plant_key)
date_key	INT FOREIGN KEY → dim_date(date_key)
time_key	INT FOREIGN KEY → dim_time(time_key)
inside_temperature	DECIMAL(5,2) NOT NULL
inside_humidity	DECIMAL(5,2) NOT NULL
outside_temperature	DECIMAL(5,2) NOT NULL
outside_humidity	DECIMAL(5,2) NOT NULL
soil_moisture	DECIMAL(5,2) NOT NULL
humidity_diff	DECIMAL(6,2) COMPUTED PERSISTED
temp_diff	DECIMAL(6,2) COMPUTED PERSISTED
wilt_confidence	DECIMAL(4,3) CHECK (0.0 <= wilt_confidence <= 1.0)
is_wilted	BIT NOT NULL DEFAULT 0
load_timestamp	DATETIME2 DEFAULT SYSUTCDATETIME()

Dimension Table: dim_plant

Column	Data Type
plant_key	INT IDENTITY(1,1) PRIMARY KEY
plant_id	INT NOT NULL UNIQUE (1, 2, 3)
plant_name	NVARCHAR(100) — e.g., 'Plant 1'
species	NVARCHAR(100)
greenhouse_zone	NVARCHAR(50)

9.3 Load Pattern

Pattern	Detail
Load Type	Incremental Upsert (MERGE statement)
Staging Table	stg_sensor_readings — truncated each ETL run
Merge Key	(plant_id, timestamp) — prevents duplicate inserts
Transaction Handling	Wrapped in explicit BEGIN/COMMIT with ROLLBACK on Python exception

10. Layer 8 — Power BI Dashboard

10.1 Data Connection

Property	Detail
Connection Type	DirectQuery to SQL Server (for near-real-time refresh)
Server	Configured via Power BI Gateway (On-premises Data Gateway)
Authentication	Windows Integrated Auth via service account
Dataset Refresh	Triggered by Airflow after each successful ETL run
Row-Level Security	Applied per greenhouse zone (not plant-level in current version)

10.2 SQL Query

```
SELECT timestamp,
       plant_name,
       status,
       confidence_score_pct,
       moisture_sensor,
       inside_temperature,
       inside_humidity,
       outside_temperature,
       outside_humidity,
       (inside_humidity - outside_humidity) AS humidity_difference
       (outside_temperature - inside_temperature) AS temperature_difference
FROM plant_monitoring;
```

10.3 DAX Measures

```
-- Wilt Risk Score
Wilt Risk Score =
VAR WiltedRate = [Wilted %]
VAR AvgMoisture = AVERAGE(Query1[moisture_sensor])
RETURN
WiltedRate * 0.6 + (1 - AvgMoisture / 100) * 4
```

```
-- Wilted % across all plants
Wilted % =
  DIVIDE(
       CALCULATE (
                 COUNTROWS(Query1),
                 Query1[status] = "Wilted"
                 ),
                 COUNTROWS(Query1)
)
```

10.4 Dashboard Visuals Inventory

Visual Title	Chart Type	Description
Wilt Risk Score	Gauge Chart	Real-time composite risk from 0.00 to 1.00
Plant Status Cards	KPI Cards	Green/amber/red status per plant (Plant 1–3)
Confidence Score Trend	Line Chart (ribbon)	Per-plant ML confidence over days 5–10
Healthy vs Wilted Count	Grouped Bar Chart	Stacked bar per plant showing healthy vs wilted readings
Temperature & Humidity	Combo Chart	Dual-axis with bar (humidity) and line (temperature)
Avg Moisture Per Plant	Multi-line Chart	Soil moisture trend lines per plant over time
KPI Cards (6 metrics)	Card Visuals	Humidity Diff, Wilted %, Temp Diff, AVG Temp, Readings, Moisture

11. Dashboard Observations & Insights

Based on the current dataset (Days 5–10, 3,984 total readings), the following key observations were identified from the Power BI dashboard:

11.1 Wilt Risk Assessment

Current Wilt Risk Score: 0.13 out of 1.00 — Low Risk. All three plants are currently classified as HEALTHY. Wilted % is 0.04%, indicating only isolated transient wilting events.

11.2 Confidence Score Trends

- Days 5–8: Confidence scores for all plants declined from ~88% to ~84.5%, suggesting improving conditions or reduced stress
- Day 8: All plants reached minimum confidence scores simultaneously (~84.5%), correlated with lowest moisture readings for Plant 2 (~18%)
- Days 8–10: Sharp upward recovery to ~90% confidence, aligned with moisture recovery across all plants
- Plant 1 consistently shows highest confidence scores; Plant 2 shows highest variance

11.3 Environmental Conditions

- Inside temperature averaged 33.52°C with a temp differential of +1.04°C above outside — indicating mild greenhouse heating effect
- Humidity differential of +7.40% inside vs outside suggests adequate atmospheric moisture for current plant health
- Day 6 anomaly: Plant 2 moisture dropped sharply to ~18% — likely an irrigation gap or sensor drift event requiring investigation
- Outside temperature showed a warming trend from Days 8–10 (increasing from ~32°C to ~35°C), tracked by inside sensors

11.4 Recommendations

- **Automate irrigation trigger when soil moisture for any plant falls below 30% for 3 consecutive readings**
- **Investigate Plant 2 Day 6 moisture anomaly — cross-reference irrigation logs**
- **Add a VPD (Vapor Pressure Deficit) alert when VPD exceeds 1.5 kPa for sustained periods**
- **Consider adding a fourth sensor for ambient CO2 to improve wilting prediction accuracy**

12. Deployment & Infrastructure

12.1 Infrastructure Summary

Component	Hosting / Environment
ESP32 Firmware	On-device (flashed via Arduino IDE / PlatformIO)
Google Sheets + Apps Script	Google Workspace (cloud-hosted, no server required)
Google Sheets API	Google Cloud Platform project with enabled API
Apache Airflow	Docker Compose on Ubuntu 22.04 server (or Astronomer.io)
Python ETL	Runs within Airflow worker containers (Python 3.11)
SQL Server	Microsoft SQL Server 2022 on-premises / Azure SQL Database
Power BI	Power BI Service (cloud) + On-premises Data Gateway

12.2 Environment Variables & Secrets

- GOOGLE_SHEETS_SPREADSHEET_ID — stored in Airflow Variable
- GOOGLE_SERVICE_ACCOUNT_KEY — stored in Airflow Connection (JSON)
- SQL_SERVER_CONNECTION_STRING — stored in Airflow Connection
- POWERBI_DATASET_REFRESH_URL — stored in Airflow Variable
- ALERT_EMAIL_RECIPIENTS — stored in Airflow Variable

12.3 Disaster Recovery

Scenario	Recovery Procedure
ESP32 failure	Replace unit; firmware re-flashed from git; data gap tolerated up to 24h
Airflow DAG failure	Auto-retry (3x); manual trigger via Airflow UI after investigation
Google Sheets API quota exceeded	Backoff + retry; quota increase request via GCP Console
SQL Server unavailable	ETL paused; Airflow retries; data buffered in staging CSV
Power BI refresh failure	Dashboard serves cached data; alert sent to owner

13. Glossary

Term	Definition
DAG	Directed Acyclic Graph — Airflow's representation of a data pipeline as nodes (tasks) and edges (dependencies)
DHT22	Digital humidity and temperature sensor by Aosong Electronics
ESP32	Wi-Fi and Bluetooth enabled microcontroller by Espressif Systems
ETL	Extract, Transform, Load — the process of moving and processing data between systems
NVS	Non-Volatile Storage — ESP32 flash memory partition for persistent key-value configuration
SCD	Slowly Changing Dimension — strategy for managing historical data in dimension tables
Wilt Risk Score	Composite metric (0.0–1.0) representing the current probability of plant wilting
XCom	Cross-Communication — Airflow mechanism for passing small data artifacts between tasks