Water Utility Performance Dashboard

User Manual

December 11, 2025

Contents

1 Introduction

This document provides a detailed guide to the metrics, visualizations, and operational workflows of the Water Utility Performance Dashboard. It is designed to empower stakeholders—from executives to field managers—with the knowledge to leverage data for decision-making.

2 User Guide

2.1 Data Import & Management

To ensure the dashboard reflects the latest operational reality, users can import data directly.

- **Navigate**: Go to any dashboard page (e.g., *Service Quality*, *Finance*).
- **Expand**: Click the **Data Import** expander at the top of the page.
- **Methods**:
  - **Upload Custom Files**: Select the "Upload Custom Files" tab. Drag and drop your `.csv` or `.xlsx` files into the respective drop zones (e.g., "Service Data", "Billing Data").
    - *Validation*: The system automatically checks for required columns and warns you if fields are missing.
  - **Default Data**: Use the "Default Data" tab and click **Reload Default Data** to reset the view to the system's baseline dataset.

2.2 Using Filters

The dashboard offers standardized filtering to drill down into specific areas. **Location Filters:**

- **Country**: Select a specific country (e.g., "Uganda") or "All". *Note: Your access may be restricted based on your user role.*
- **Zone/City**: Choose specific operational zones or cities. On the **Production** page, this is a multi-select field allowing you to compare multiple zones simultaneously.

Time Filters:

- **Period**: Toggle between "Annual", "Quarterly", "Monthly", or "Daily" views (availability depends on the specific dashboard page).
- **Year/Month**: Use the dropdowns to focus on a specific timeframe.

2.3 Exporting Data

CSV Export: Look for the **"Export CSV"** button in the top-right header of each dashboard page. This downloads the currently filtered dataset for further analysis in Excel or other tools.

2.4 AI Assistant (MajiBot)

MajiBot is an AI-powered analyst integrated into the dashboard.

- **Access**: Click the **"Ask MajiBot"** button or look for the chat icon (if enabled for your role).
- **Capabilities**:
  - **Data Queries**: Ask natural language questions like *"What is the NRW rate for Kampala?"* or *"Show me the collection efficiency trend."*

  - **Insights**: MajiBot analyzes the current context (filters, anomalies) to provide relevant answers.

- **Configuration**:
  - Managers can configure the AI provider (Gemini or Grok) in the settings.
  - **API Keys**: Ensure a valid API key is set in `.streamlit/secrets.toml` or environment variables to enable this feature.

3 Detailed Metric Calculations

3.1 Executive Dashboard

3.1.1 Service Coverage Score

Mathematical Formula:

$$\text{Coverage Score} = \frac{\text{Water Safely Managed\%} + \text{Sanitation Safely Managed\%}}{2} \tag{1}$$

Python Implementation:

```
water_safely_managed = df_water['w_safely_managed_pct'].mean()
sanitation_safely_managed = df_sewer['s_safely_managed_pct'].mean()
coverage_score = (water_safely_managed + sanitation_safely_managed) / 2
```

Status Thresholds: >80% | 60-80% | <60%

3.1.2 Financial Health Index

Mathematical Formula:

$$\text{Financial Score} = (C_{eff} \times 0.4) + (O_{cov} \times 0.4) + (B_{util} \times 0.2) \tag{2}$$

Where:

- C_{eff} = Collection Efficiency (capped at 100%)
- O_{cov} = Operating Coverage (capped at 120%)
- B_{util} = Budget Utilization (capped at 100%)

Python Implementation:

```
collection_efficiency = min((total_paid / total_billed) * 100, 100)
operating_coverage = min((total_revenue / total_opex) * 100, 120)
budget_utilization = min((total_opex / total_budget_allocated) * 100,
    100)
financial_score = (collection_efficiency * 0.4) + (operating_coverage *
     0.4) + (budget_utilization * 0.2)
```

3.1.3 Operational Efficiency Score

Mathematical Formula:

$$\text{Operational Efficiency} = \frac{(100 - \text{NRW\%}) + \text{Cap. Util\%} + \text{Service Hours Normalized}}{3} \tag{3}$$

Python Implementation:

```
nrw_percent = ((production_m3 - consumption_m3) / production_m3) * 100
capacity_utilization = (wastewater_treated / wastewater_capacity) * 100
service_hours_normalized = (avg_service_hours / 24) * 100
operational_efficiency = ((100 - nrw_percent) + capacity_utilization +
    service_hours_normalized) / 3
```

3.1.4 Service Quality Index

Mathematical Formula:

$$\text{Quality Index} = \frac{\text{WQ Compliance\%} + \text{Resolution Rate\%} + \text{Asset Health}}{3} \tag{4}$$

Python Implementation:

```
wq_compliance = (tests_passed / tests_conducted) * 100
customer_resolution = (complaints_resolved / total_complaints) * 100
asset_health = df_national['asset_health'].mean()
quality_index = (wq_compliance + customer_resolution + asset_health) /
    3
```

3.2 Access & Coverage Dashboard

3.2.1 Water Supply Coverage

Mathematical Formula:

$$\text{Coverage\%} = \frac{\text{Municipal Coverage Count}}{\text{Total Population}} \times 100 \tag{5}$$

Python Implementation:

```
total_population = df_water['popn_total'].sum()
municipal_coverage_count = df_water['municipal_coverage'].sum()
municipal_supply_pct = (municipal_coverage_count / total_population) *
    100 if total_population > 0 else 0
```

3.2.2 Year-over-Year Growth

Mathematical Formula:

$$\text{YoY Growth} = \text{Coverage}_{current} - \text{Coverage}_{previous} \tag{6}$$

Python Implementation:

```
last_year_data = df_water[df_water['year'] == selected_year - 1]
last_year_pct = (last_year_data['municipal_coverage'].sum() /
    last_year_data['popn_total'].sum()) * 100
yoy_growth = municipal_supply_pct - last_year_pct
```

3.2.3 Sewer Coverage (Quarterly)

Python Implementation:

```
df_service['quarter'] = ((df_service['month'] - 1) // 3) + 1
df_q_latest = df_service[df_service['quarter'] == df_service['quarter']].max()]
total_sewer_conn = df_q_latest['sewer_connections'].mean()
total_households = df_q_latest['households'].max()
sewer_coverage_pct = (total_sewer_conn / total_households) * 100 if total_households > 0 else 0
```

3.3 Service Quality Dashboard

3.3.1 Water Quality Compliance

Mathematical Formula:

$$\text{Compliance}\% = \frac{\text{Tests Passed}_{Chlorine} + \text{Tests Passed}_{E.coli}}{\text{Tests Conducted}_{Chlorine} + \text{Tests Conducted}_{E.coli}} \times 100 \tag{7}$$

Python Implementation:

```
chlorine_passed = df_service['test_passed_chlorine'].sum()
chlorine_conducted = df_service['tests_conducted_chlorine'].sum()
ecoli_passed = df_service['tests_passed_ecoli'].sum()
ecoli_conducted = df_service['test_conducted_ecoli'].sum()
total_passed = chlorine_passed + ecoli_passed
total_conducted = chlorine_conducted + ecoli_conducted
compliance_rate = (total_passed / total_conducted) * 100 if total_conducted > 0 else 0
```

3.3.2 Network Performance (Blockages)

Mathematical Formula:

$$\text{Blockages per 100km} = \frac{\text{Total Blockages}}{\text{Total Sewer Length}} \times 100 \tag{8}$$

Python Implementation:

```
total_blockages = df_fin['blocks'].sum()
# Avoid double-counting monthly data
total_sewer_length = df_fin.groupby('city')['sewer_length'].max().sum()
blockages_per_100km = (total_blockages / total_sewer_length * 100) if total_sewer_length > 0 else 0
```

3.4 Financial Health Dashboard

3.4.1 Collection Efficiency

Mathematical Formula:

$$\text{Collection Efficiency}\% = \frac{\text{Revenue Collected}}{\text{Amount Billed}} \times 100 \tag{9}$$

Python Implementation:

```
collection_rate = (sewer_revenue / sewer_billed) * 100
# Aggregate
total_billed = df_fin_service['sewer_billed'].sum()
total_revenue = df_fin_service['sewer_revenue'].sum()
avg_collection_rate = (total_revenue / total_billed) * 100
```

Target: >85% (Green) | 60-85% (Yellow) | <60% (Red)

3.4.2 Outstanding Debt

Mathematical Formula:

$$\text{Debt} = \text{Amount Billed} - \text{Revenue Collected} \tag{10}$$

Python Implementation:

```
debt_per_record = sewer_billed - sewer_revenue
total_debt = df_fin_service['debt'].sum()
debt_to_billed_ratio = (total_debt / total_billed) * 100
```

3.4.3 Cost Recovery Ratio

Mathematical Formula:

$$\text{Cost Recovery\%} = \frac{\text{Total Revenue}}{\text{Operational Expenses (OPEX)}} \times 100 \tag{11}$$

Python Implementation:

```
cost_recovery_ratio = (sewer_revenue / opex) * 100
```

Interpretation: >100% indicates financial self-sufficiency

3.4.4 Revenue per Staff

Mathematical Formula:

$$\text{Revenue per Staff} = \frac{\text{Total Revenue}}{\text{Sanitation Staff} + \text{Water Staff}} \tag{12}$$

Python Implementation:

```
total_staff = sanitation_staff + water_staff
revenue_per_staff = sewer_revenue / total_staff
```

3.5 Production & Operations Dashboard

3.5.1 Capacity Utilization

Mathematical Formula:

$$\text{Capacity Util\%} = \frac{\text{Actual Production}}{\text{Estimated Design Capacity}} \times 100 \tag{13}$$

Where: Estimated Capacity = max(Production per Source) × 1.1 **Python Implementation:**

```
latest_date = df_production['date_dt'].max()
df_latest = df_production[df_production['date_dt'] == latest_date]
total_production_latest = df_latest['production_m3'].sum()
estimated_capacity_per_source = df_production.groupby('source')['
    production_m3'].max() * 1.1
total_estimated_capacity = estimated_capacity_per_source.sum()
capacity_utilization = (total_production_latest /
    total_estimated_capacity) * 100 if total_estimated_capacity > 0 else
     0
```

3.5.2 Non-Revenue Water (NRW)

Mathematical Formula:

$$\text{NRW\%} = \frac{\text{Production} - \text{Consumption}}{\text{Production}} \times 100 \tag{14}$$

Python Implementation:

```
production_total = df_production['production_m3'].sum()
consumption_total = df_billing['consumption_m3'].sum()
nrw_percent = ((production_total - consumption_total) /
    production_total) * 100
```

Classification:

- **Excellent:** <15%
- **Good:** 15-25%
- **Fair:** 25-40%
- **Poor:** >40%

3.5.3 Resource Extraction Rate

Mathematical Formula:

$$\text{Extraction Rate\%} = \frac{\text{Annual Production}}{\text{Renewable Resources}} \times 100 \tag{15}$$

Python Implementation:

```
total_annual_production = df_production['production_m3'].sum()
estimated_resources = total_annual_production * 1.45  # Heuristic
extraction_rate = (total_annual_production / estimated_resources) * 100
```

Sustainability Thresholds:

- **Sustainable:** <70%
- **Monitor:** 70-90%
- **Critical:** >90%

4 Administration

4.1 Changing Passwords

(Restricted to Admin Users)

1. Navigate to **Admin Panel** (Sidebar > Admin)
2. Select user from "Manage Users" list
3. Enter new password in "Reset Password" field
4. Click **Update**

4.2 API Key Management

Enable AI features by configuring `.streamlit/secrets.toml`:

```
[llm]
GEMINI_API_KEY = "your-key-here"
# OR
GROK_API_KEY = "your-key-here"
```

Restart dashboard to apply changes.