Pharmaceutical Tablet Plant Example

Simulated pharmaceutical manufacturing process with end-to-end data historization from FrameworX PostgreSQL Snowflake.

Updated: March 2026 | **FrameworX:** fx-2020.1.3+

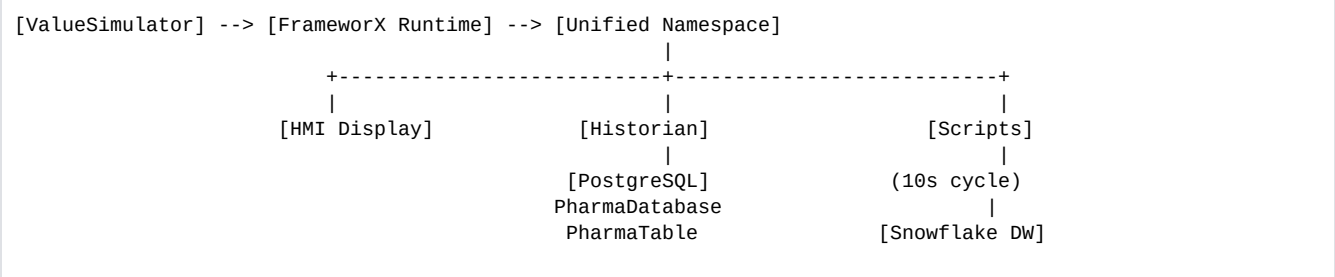
> **Download**
>
> Download PharmaTablet.dbsln
> No hardware required — uses ValueSimulator to generate all process data.

Video Tutorial — Tag Configuration in FrameworX

Overview

The **PharmaTablet** example demonstrates a complete pharmaceutical tablet manufacturing plant built in FrameworX, covering three production stages: **granulation**, **tablet compression**, and **film coating**. Simulated process data is collected in real time, historized to PostgreSQL, and synchronized to Snowflake every 10 seconds — showcasing a full OT/IT data pipeline with no custom middleware.

Architecture



Unified Namespace (UNS)

All tags are organized under an ISA-95 compliant **Unified Namespace** — a single logical hierarchy that decouples data consumers (HMI, Historian, Scripts) from the underlying device layer. Replacing the simulator with a real PLC requires only a device binding change; nothing else in the solution needs to be updated.

UNS Path	Equipment	Key Variables
`TabletPlant/Granulation/Granulator_01`	High-Shear Granulator	Temperature, Impeller Speed, Chopper Speed, Moisture, Phase, Batch ID
`TabletPlant/Compression/Press_01`	Tablet Press	Turret Speed, Compression Force, Tablet Weight, Thickness, Production Count, Reject Count
`TabletPlant/Coating/CoatingPan_01`	Film Coating Pan	Inlet/Outlet Temp, Pan Speed, Spray Rate, Atomization Pressure, Coat Weight Gain

Historian — PostgreSQL

All process variables are continuously historized via the FrameworX **Historian** module into a PostgreSQL database (`PharmaDatabase`, table `PharmaTable`). The Historian handles timestamping, compression, and deadband filtering automatically.

> **Configuration Required**
>
> PostgreSQL credentials must be set in **Datasets > DBs** before starting the runtime.

Snowflake Synchronization

A server-side script (`write_postgre_to_snowflake`) runs on a **10-second cycle**, reading new rows from PostgreSQL and writing them incrementally to **Snowflake**. This creates a near-real-time bridge to the cloud data warehouse for analytics and reporting workloads.

Property	Value
Sync interval	10 seconds
Source	PostgreSQL · `PharmaDatabase.PharmaTable`
Destination	Snowflake (configurable)
Sync mode	Incremental — new rows only

> **Configuration Required**
>
> Snowflake credentials (account, warehouse, database, schema, user, password) must be set in **Datasets > DBs** before starting the runtime.

HMI Overview

The **MainPage** display shows a plant-wide process overview in a single screen, organized left to right following the physical production flow.

Zone	Symbol	Gauges
Granulator	Animated blower (impeller motion)	SemiCircle — Temperature; Linear — Moisture (color-ranged)
Tablet Press	Tacho — Turret Speed	CircularGauge — Compression Force; KPI boxes — Production / Rejects
Coating Pan	Tacho — Pan Speed	Dual vertical linear gauges — Inlet / Outlet Temperature

Two live **TrendCharts** at the bottom track Press parameters (Compression Force, Weight, Thickness) and thermal data (Granulator Temperature, Coating Inlet/Outlet Temps) over a 5-minute rolling window.

Requirements

Component	Required	Notes
FrameworX	✅	fx-2020.1.3 or later, Enterprise license
PostgreSQL	✅	Version 12+. Create `PharmaDatabase` before first run.
Snowflake Account	✅	Trial account sufficient for demo.
Physical PLC	ℹ️	Not required — ValueSimulator generates all data.

Getting Started

1. **Download and open** `PharmaTablet.dbsln` in the FrameworX Designer.
2. **Configure PostgreSQL** — update the historian connector connection string. The table `PharmaTable` is created automatically on first run.
3. **Configure Snowflake** — open the `write_postgre_to_snowflake` script and set your account, warehouse, database, schema, username, and password at the top of the file.
4. **Start the Runtime** (F5). The simulator starts immediately, historization begins, and the Snowflake sync script activates on its 10-second cycle.
5. Open **MainPage** to view the live process overview.

> **Verify the pipeline**
>
> Run `SELECT COUNT(*) FROM PharmaTable` in your Snowflake worksheet — the row count should increase every 10 seconds while the runtime is active.

In this section...