TrendMiner Success Series - Admin Track

September 9th, 2026

Connector Modernization
TrendMiner Providers & ODBC Improvements

Welcome to the webinar. We will start momentarily.

TrendMiner

Listen only mode

Use the question area in the side panel

Q&A at the end

A recording and slides will be made available after the webinar

In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

Why connectivity requires Windows

- Historian vendors ship Windows based SDKs (COM / .NET)
- ODBC drivers are only released and supported for Windows
    - JDBC drivers are not always available cross-platform
- Firewall security requirements
    - AWS security
    - Appliance is on different access level

Historians that require the Windows

- AspenTech IP21 — Windows ODBC interface
- Exaquantum — Windows .NET SDK
- Honeywell PHD — Windows .NET SDK
- Yokogawa Wonderware — Windows SDK/ODBC
- Aveva Wonderware — Windows .NET SDK/ODBC
- Aveva PI — Windows .NET SDK
- GE Proficy — Windows .NET SDK

Note: Refer documentation for full up-to-date list

In today’s session

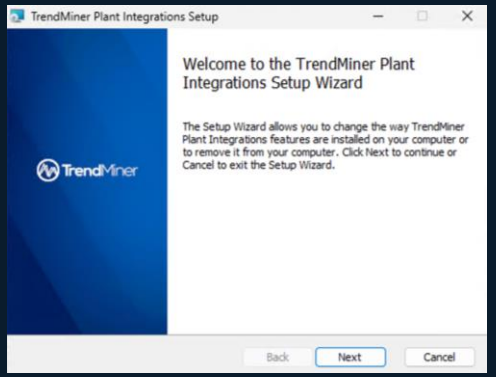
Plant Integrations	The foundation
WebHost → AppHost	Plant Integration management improvements
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

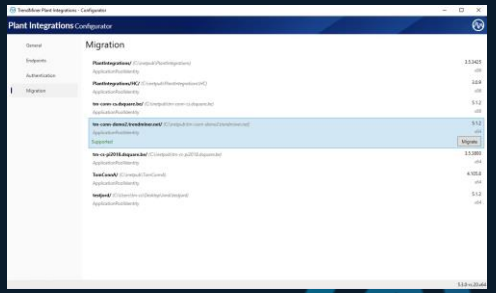
Installation and upgrades – easier and friendlier than IIS

Port 8001 should be available during installation

Local firewall rules are automatically updated

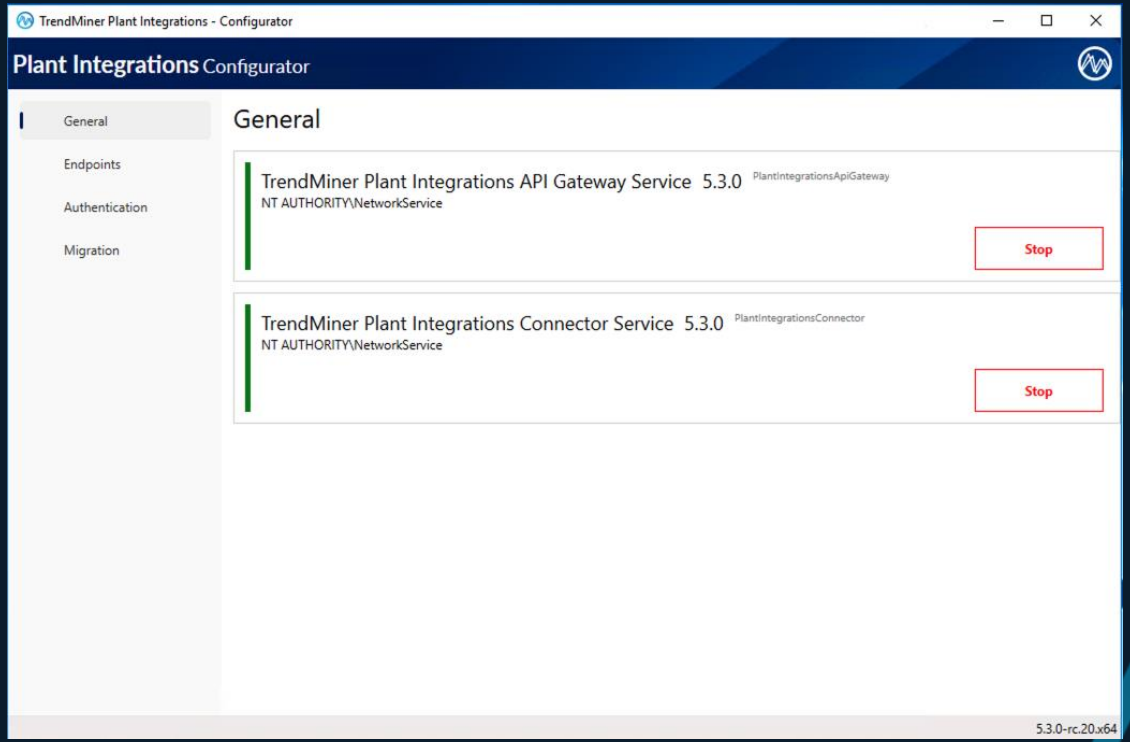
Configuration: Authentication, endpoints, certificates, services management





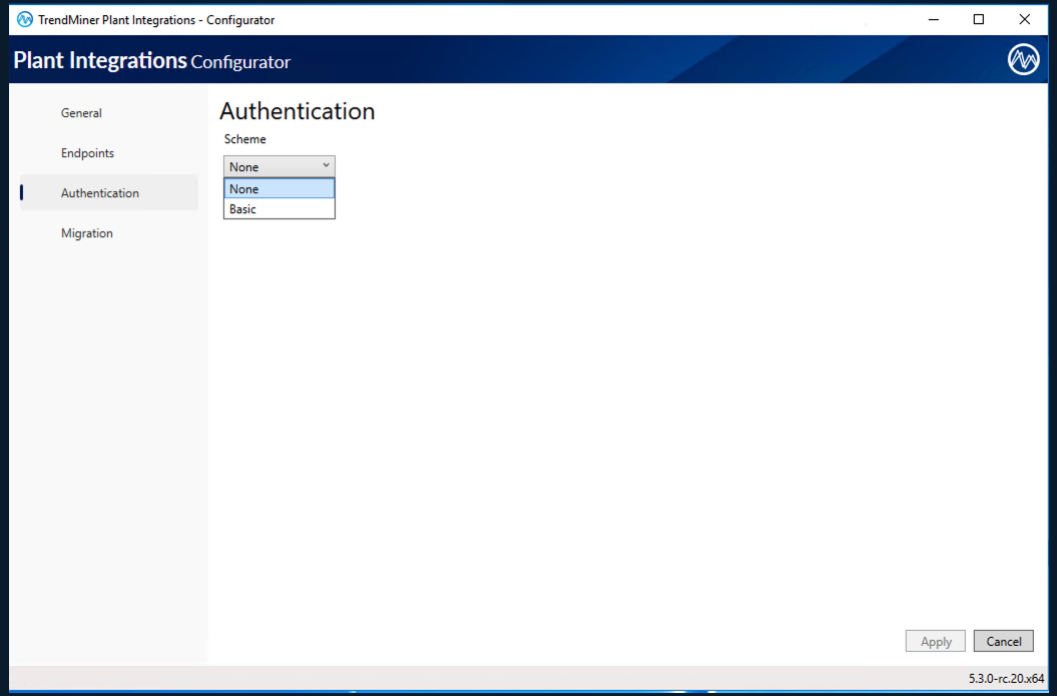
Management - easier and friendlier than IIS

General services overview



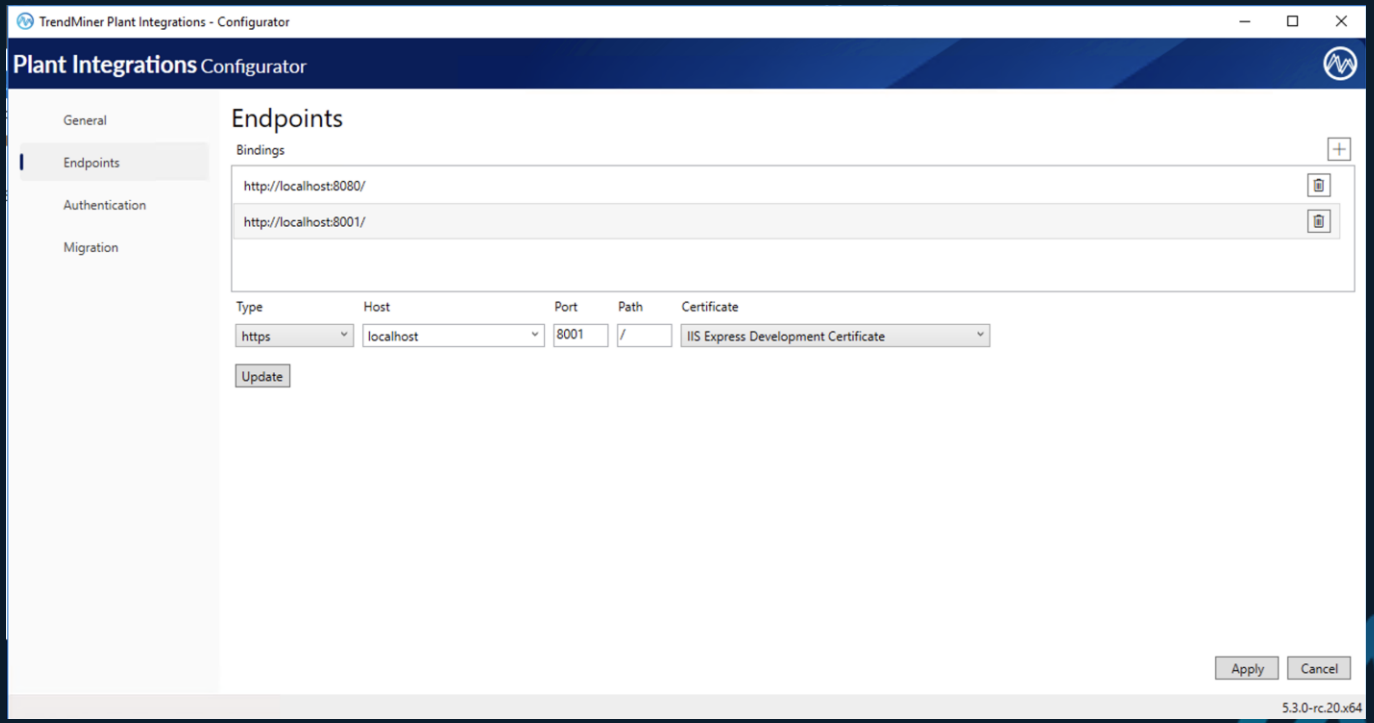
Management - easier and friendlier than IIS

Authentication Configuration



Management - easier and friendlier than IIS

Endpoints



Important Notes

Unsupported

- Currently one Plant Integration service
- Legacy ODBC, SQLite and OleDB providers do not migrate to AppHost
- Migration is supported from connector v5 only (R2025.R2 release)

Keep in mind

- After migration: update the port in ConfigHub to point to the new connector
- Fallback is possible back to IIS
- The modern replacement is Generic ODBC / JDBC

In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

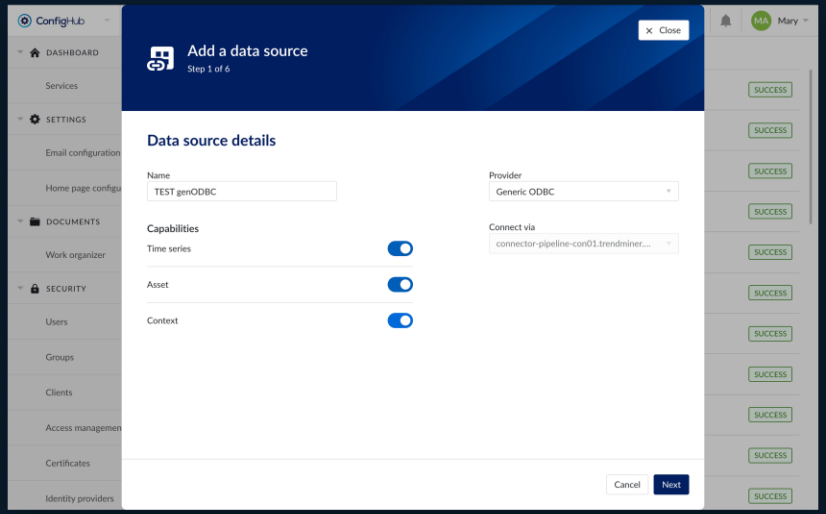
Why Legacy File Based configuration is being retired

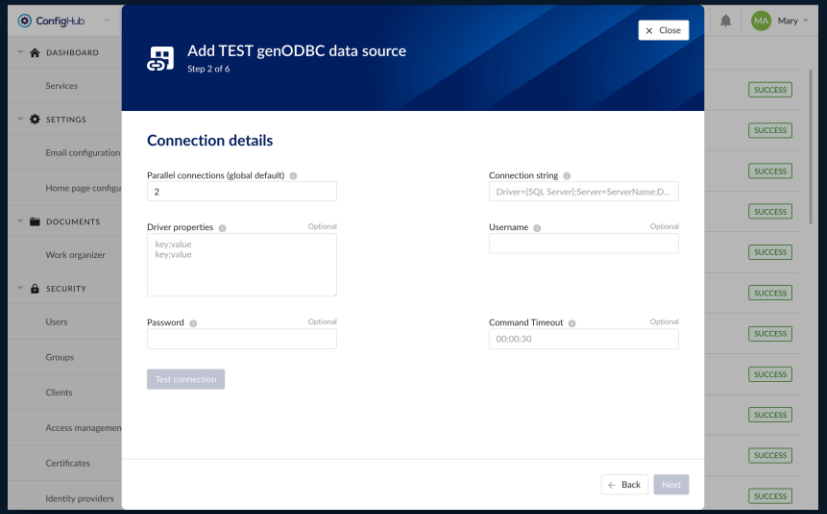
- Built long ago, its design predates today's connectivity standards
- Config and maintenance model has not kept pace with the platform
- No parity with Generic ODBC/JDBC tooling
- Legacy ODBC has become difficult to configure and maintain

Investigating Legacy Technologies Alternatives

- OleDB
- Sqlite

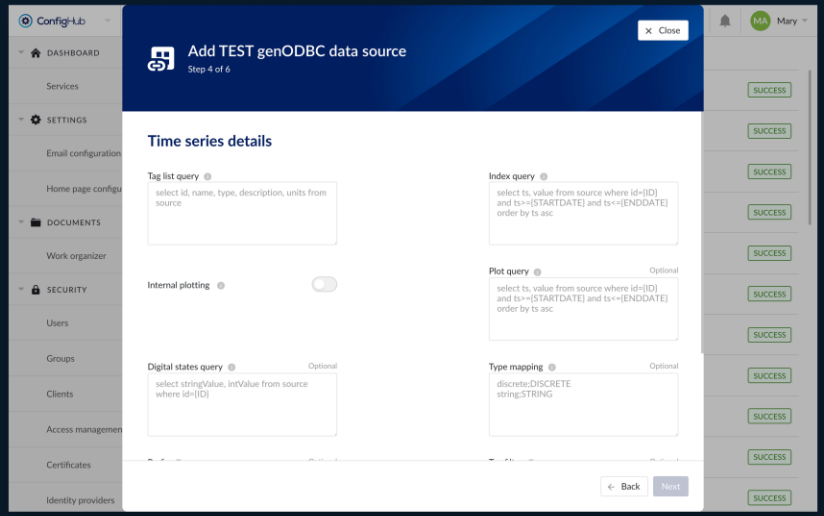
Control over Configuration

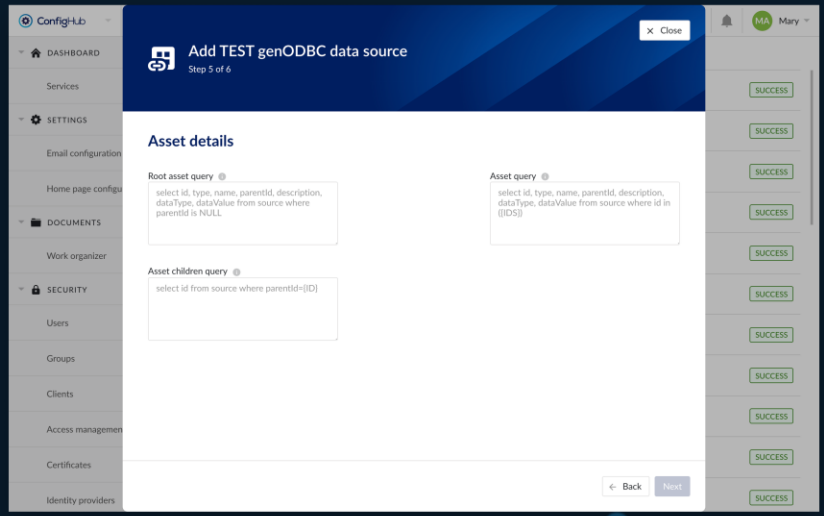




Generic ODBC/JDBC

Access to Configuration





Generic ODBC/JDBC

Generic ODBC/JDBC parameter syntax

Parameters now use meaningful names instead of positional ?/{0} markers

Note: there're small differences between ODBC and JDBC, refer documentation

Legacy ODBC

```
Legacy ODBC

• Using placeholders {0}, {1}, {2}, {3}
• Using unnamed parameters ?, ?, ?, ?
```

Positional binding — SQL injection risk when substituted as raw text

Generic ODBC / JDBC

```
Generic ODBC / JDBC

• Using named parameters
  {ID} — tag name in ODBC, id in JDBC
  {NAME} — tag name
  {INTERPOLATION} — interpolation type name
  {STARTDATE} — formatted datetime string
  {ENDDATE} — formatted datetime string
  {NUMBEROFINTERVALS} — number of
  {INDEXRESOLUTION} — index resolution
```

Named binding — no SQL injection; query must be written to rely on parameters

Generic ODBC/JDBC as an Alternative

- Ignition stores tag history in underlying database
  - MySql
  - MSSQL
  - Postgres TimescaleDB
- Wonderware
- Siemens PCS7
- AspenTech

Not just time series

- Legacy ODBC: time series data only
- Generic ODBC/JDBC and custom connectivity support all three data types:
    - Time Series — the classic case, tag values over time
    - Context — events, batches, process anomalies
    - Asset — structural and hierarchical asset data

Generic JDBC removes the middle tier

Advantages

- Middle Tier
  - Plant Integration: Historian → Plant Integrations → TrendMiner
  - JDBC: Historian → TrendMiner via JDBC driver
- No Windows machine in the middle — where a JDBC driver is available

Except

- Indirect connection still applies when no native driver exists — Windows connector is required
- AWS SaaS uses a reverse-proxy agent for private VPN tunneling and requires Plant Integration.
- Security reasons

Generic JDBC Drivers

Supplied

- MSSQL
- Postgres
- Snowflake
- Databricks
- MariaDB
- Sqlite

Third-Party

- Copy Jar to appliance

In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

Connector API: build your own integration

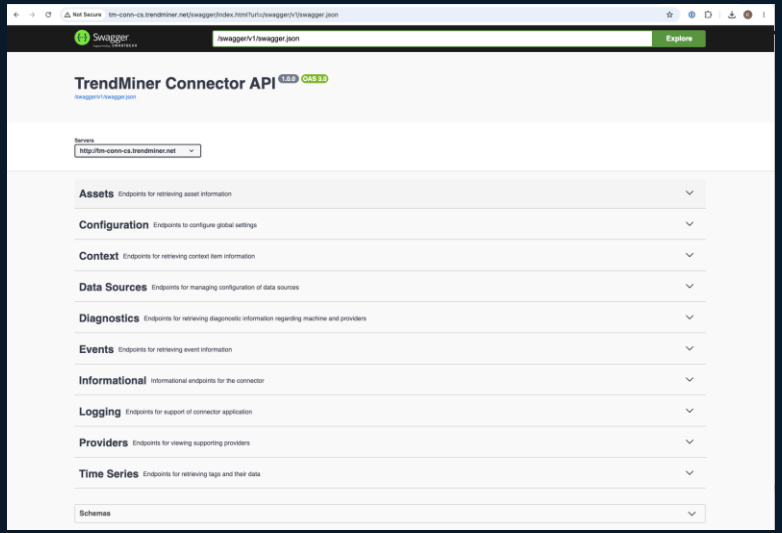
Starting Point:

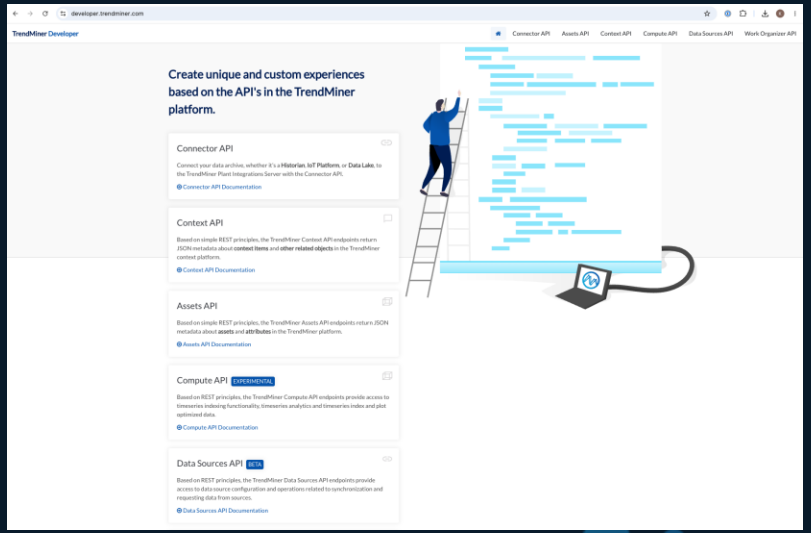
- Swagger
- Developer Portal
- Example Documentation
- Community Posts

Examples:

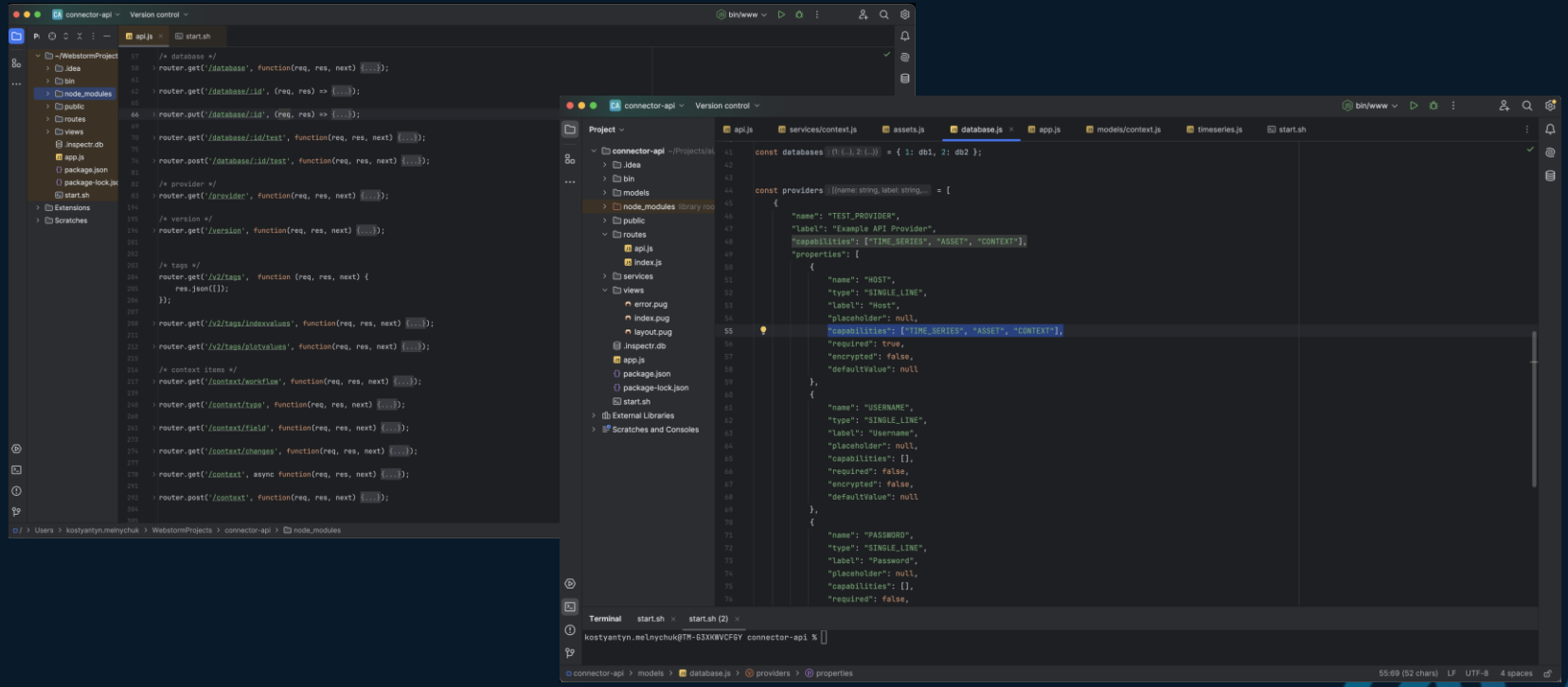
- Ignition Connectivity
  - Connector API: WebDev module usage by means of Ignition Designer (Jython)
  - Connector API: Custom Java Module option
- Litmus
- Any REST API Service, e.g. Node.js

Demo: Node.js simulated Plant Integrations API

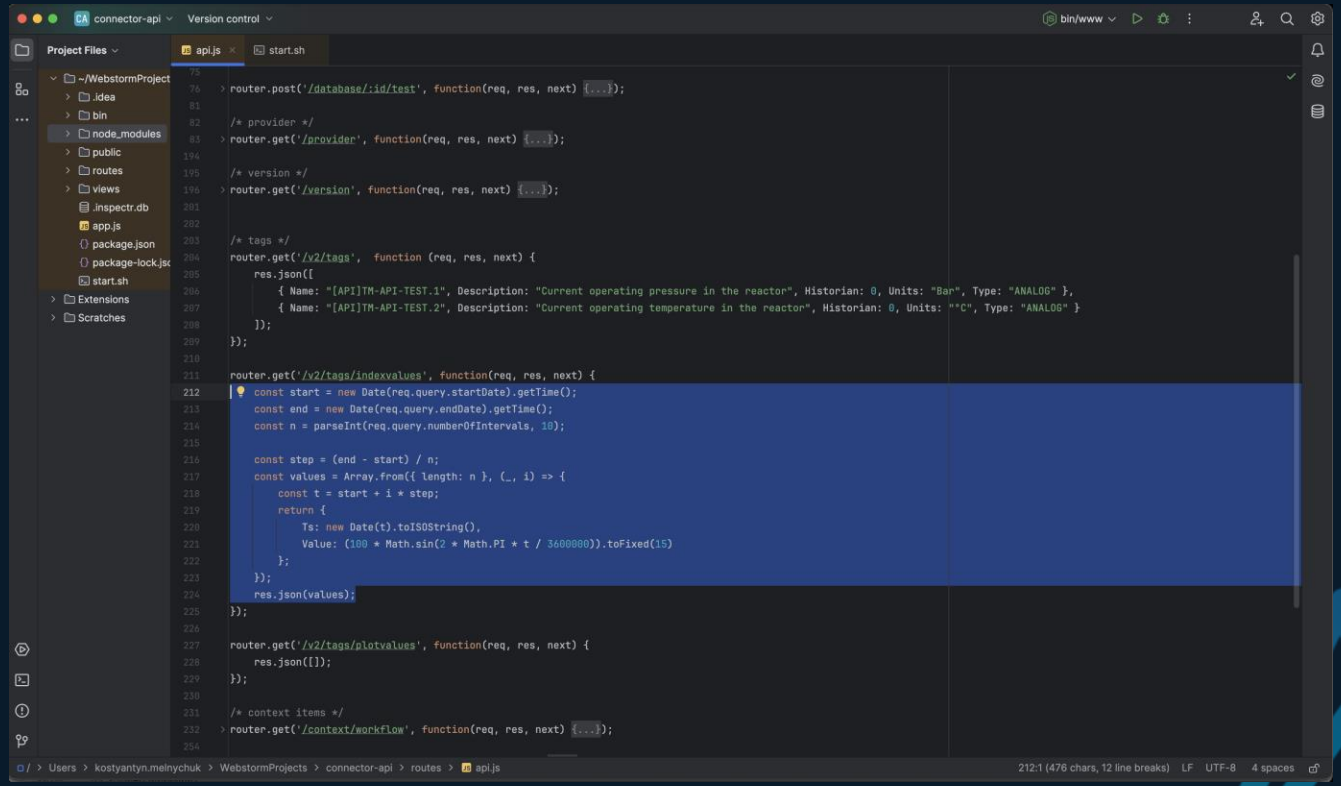




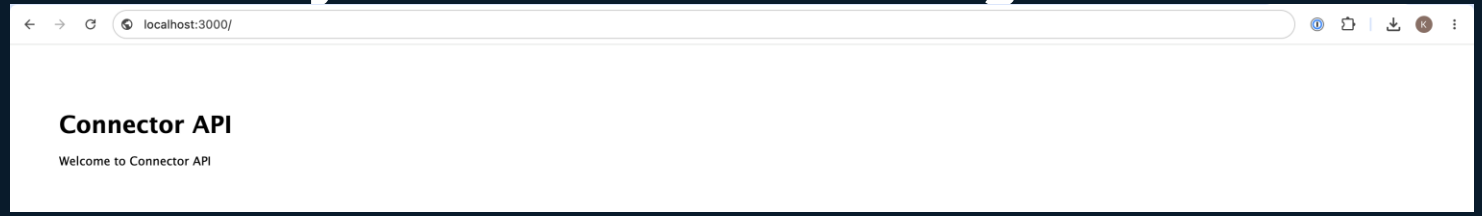
Demo: Node.js simulated Plant Integrations API

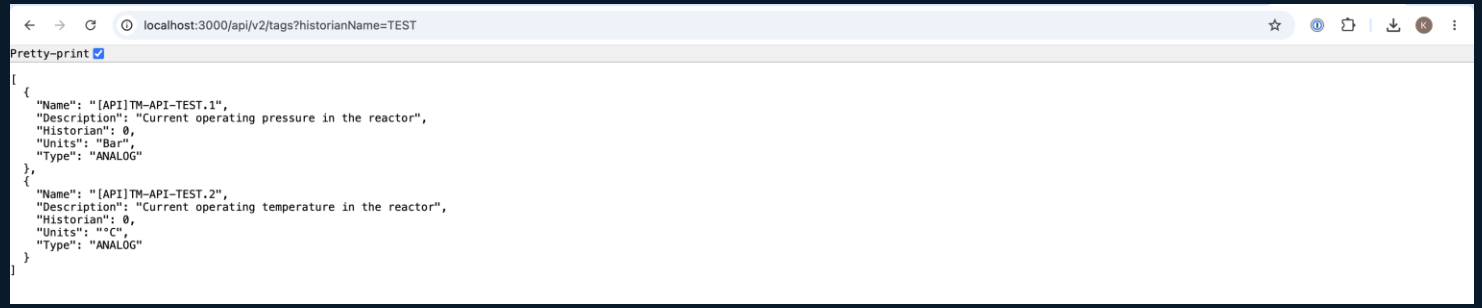


Demo: Node.js simulated Plant Integrations API

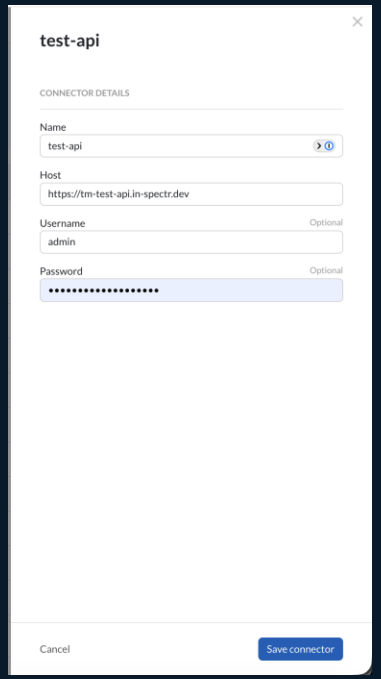


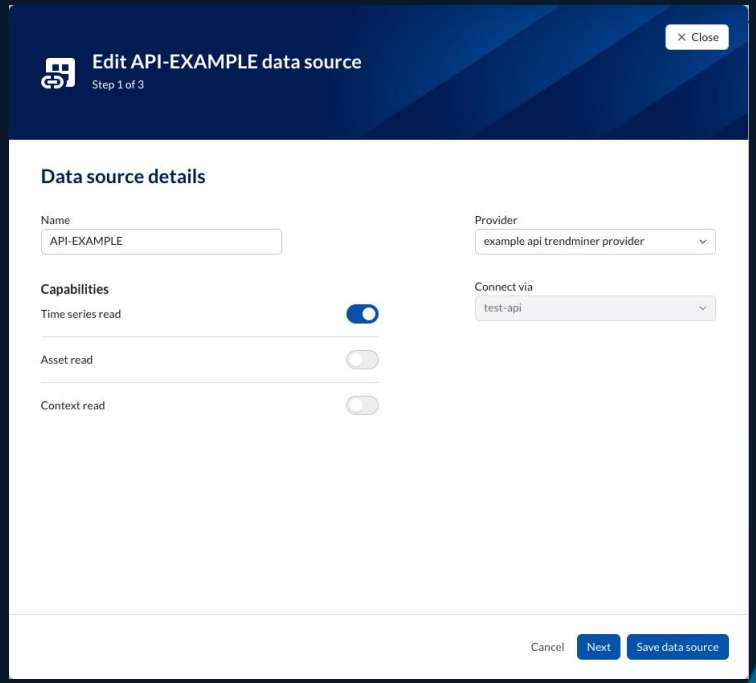
Demo: Node.js simulated Plant Integrations API



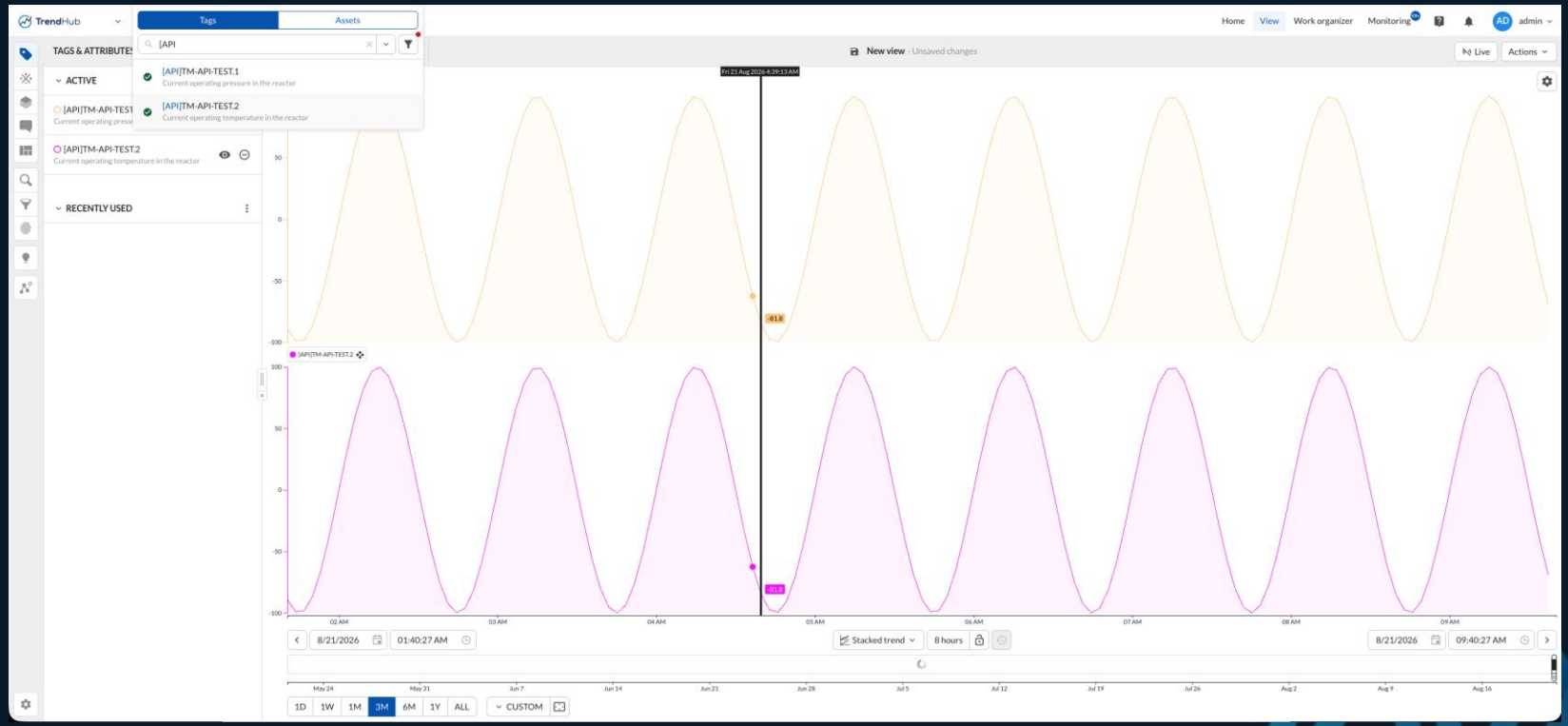


Demo: Node.js simulated Plant Integrations API



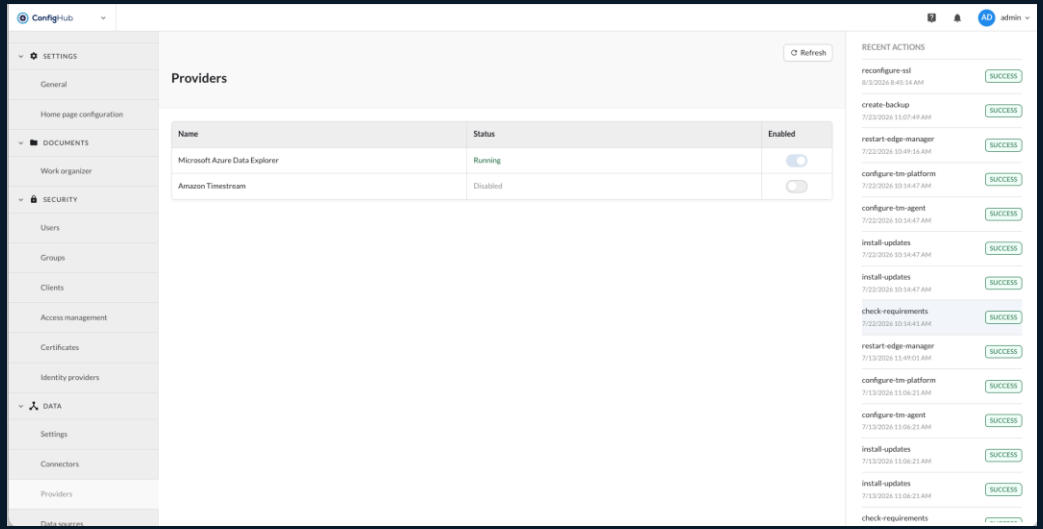


Demo: Node.js simulated Plant Integrations API



Additional built-in providers

- Canary (Built-In)
- Amazon Timestream (Runtime)
- Azure Data Explorer (Runtime)



Generic ODBC/JDBC

Need help?

TrendMiner can make an offer to investigate your connectivity options.

- 2 phases:
  - Investigation and performance report
  - Actual implementation and productization
- One-year updates included
- No cure no pay*
- Contact your CSM

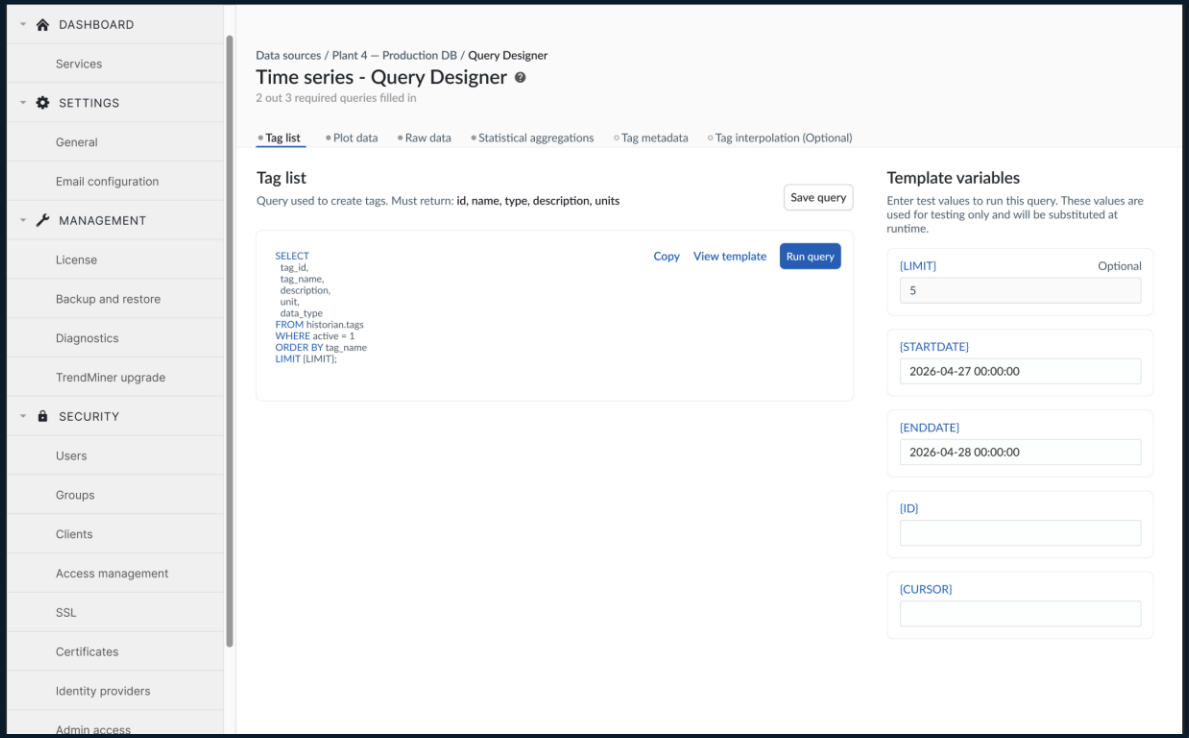
*In case the connection cannot be set up or performance of the data source is insufficient, only part of the investigation effort is charged, not the full price.

In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

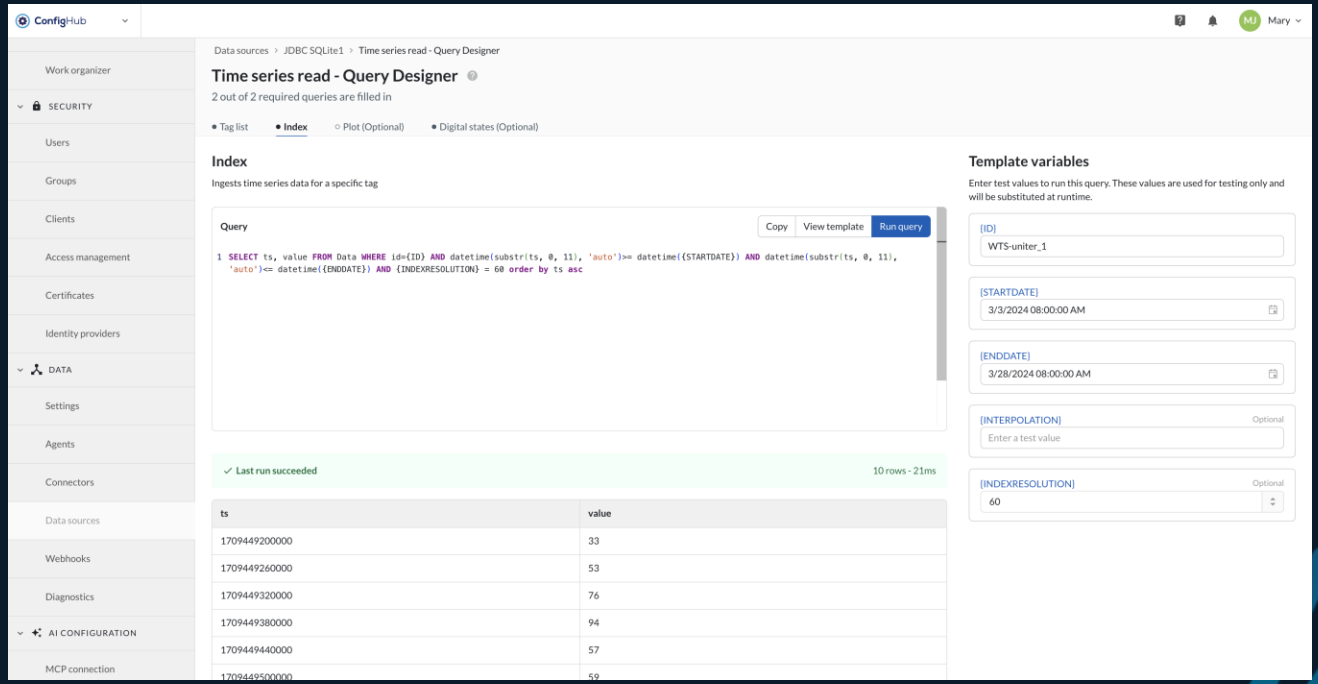
Future improvements:

Query Designer



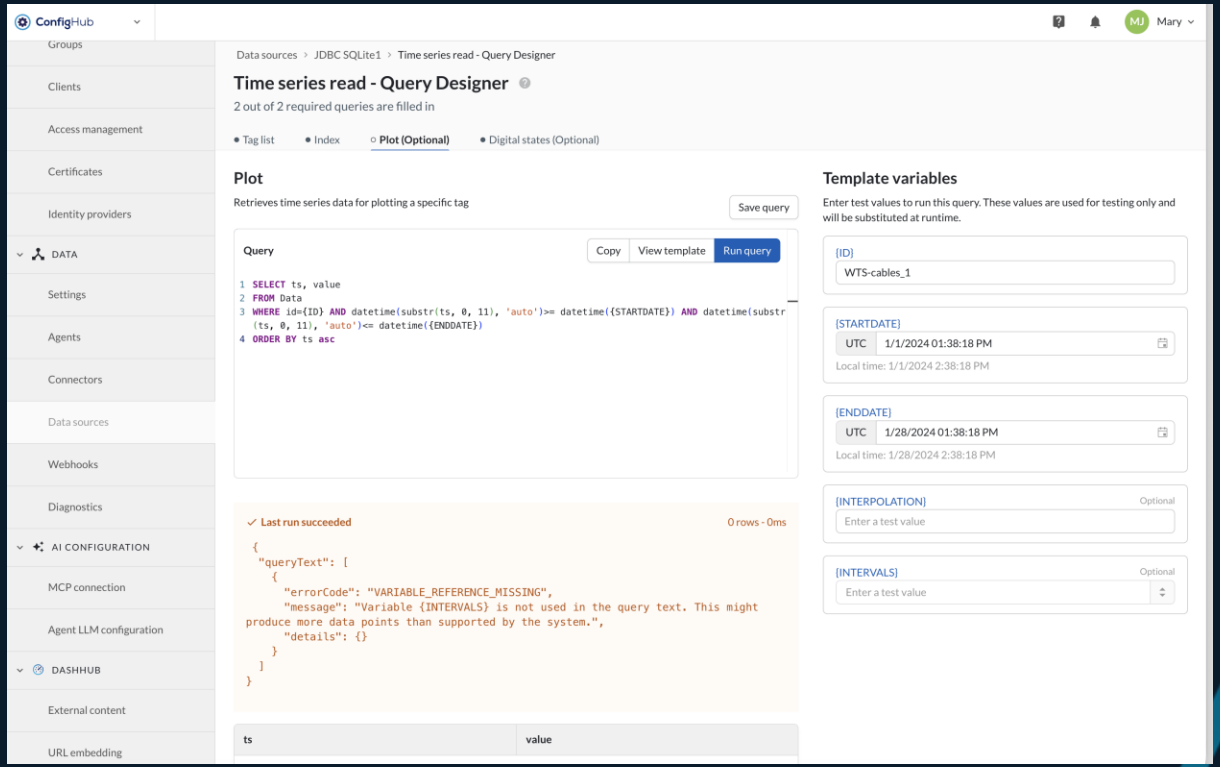
Future improvement:

Test queries



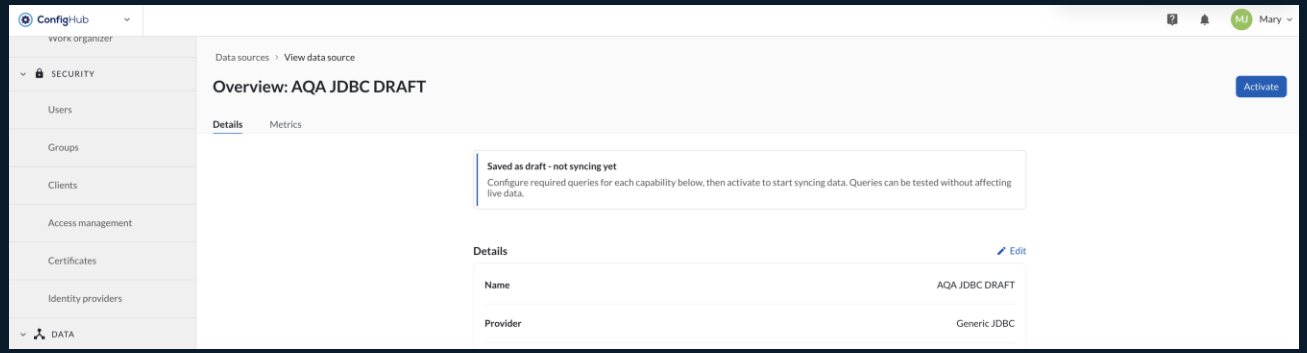
Future improvement:

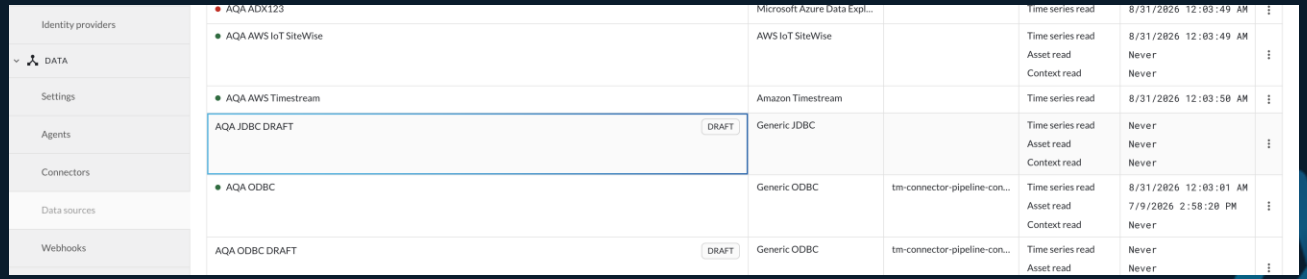
Diagnose problems



Future improvement:

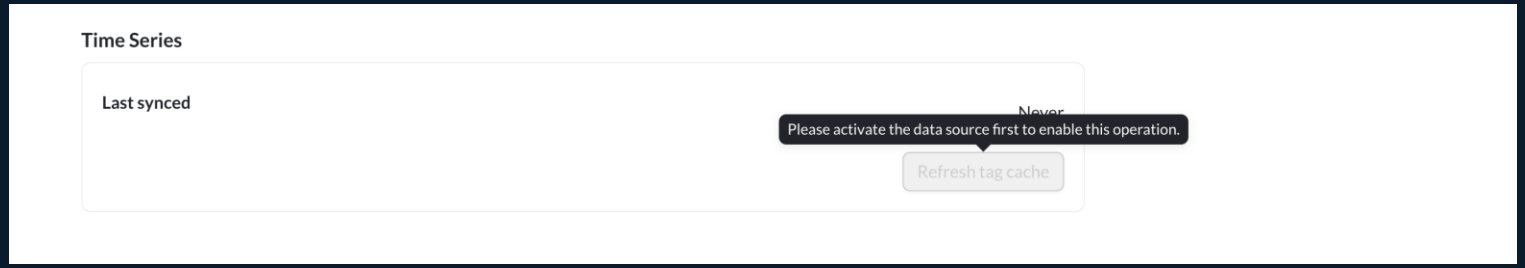
Draft concept: Active/Inactive





Future improvement:

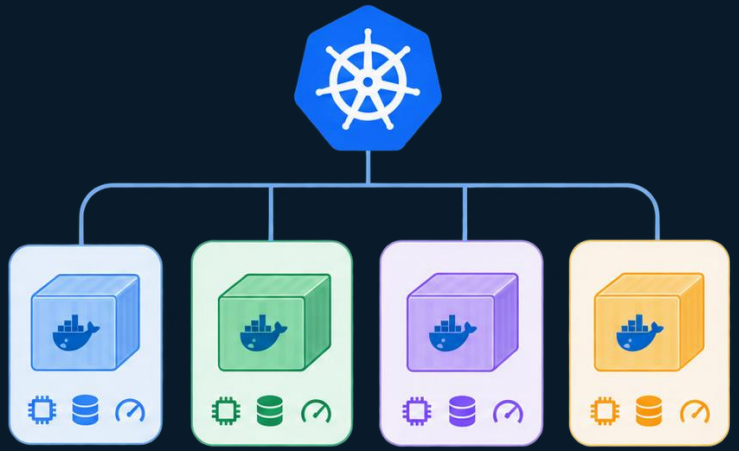
Draft concept: Active/Inactive



Future improvement:

Standalone runtimes for all providers: Stability and performance

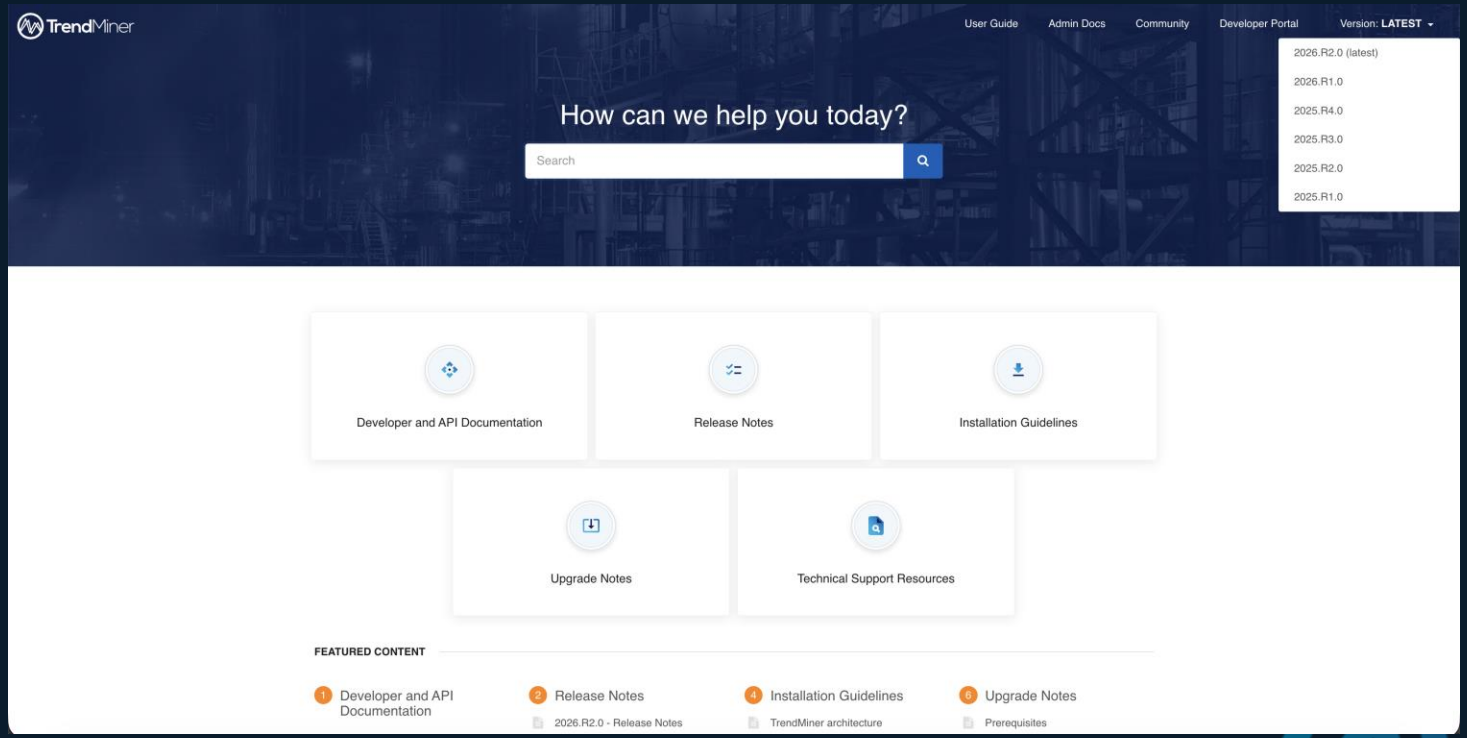
- Any failure does not impact other services
- Manages its own system resources
- Better management



In today’s session

Plant Integrations	The foundation
WebHost → AppHost	Migration path
Legacy ODBC vs Generic ODBC	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

documentation.trendminer.com



Pages referenced in this webinar

- Generic ODBC/JDBC documentation
- TrendMiner connectivity overview
- Migration WebHost → AppHost
- Supported time series connectivity which requires query definition
- How to install third party JDBC driver
- Connector API documentation

In today’s session

Why connectivity requires Windows	The foundation
WebHost → AppHost	Migration path
Legacy ODBC: what changes	New patterns & parameters
Custom connectivity	Connector API
Future improvements	What’s coming
Useful Resources	Documentation & links
Q&A	Wrap up

Questions?