On the left is my new ChatGPT-generated logo. I simply asked: “Give me a logo for my company. The n

I use AI whenever possible—whether it's writing PySpark code, drafting a text, finding information, or learning. Needless to say, it makes many things easier. We're witnessing a revolution unfold before our eyes. It boosts productivity and frees up time for deeper thinking and creativity. Tasks that once took hours can now be done in minutes with the right prompts. The line between human capability and machine support is blurring—and it's changing how we work, learn, and solve problems.

2026-05-02 OpenSource Lakehouse on Scaleway — a Saturday revelation

I've been a big fan of Azure Databricks for years, and honestly I still am. But the geopolitical reality of 2026 has forced me to seriously look at alternatives. European sovereignty, vendor lock-in, the whole conversation. I always assumed open source meant too much glue code, too much overhead, too much yak shaving. Turns out I was wrong.

One Saturday morning I rolled out a complete analytics platform on Scaleway's Managed Kubernetes (a French cloud provider) in just a few hours. Full lakehouse architecture: S3-compatible object storage as the foundation, Spark for processing, Hive Metastore for the catalog, **Apache Ranger** and **OpenMetadata** for governance, **Trino** and **ClickHouse** as query engines,
Superset
for BI,
Airflow
for orchestration, and
dbt
for transformations. End-to-end. I threw 1-billion-row files at it to see what would break. Not much did.

Of course, none of this would have been possible in a single afternoon without AI. And that's exactly my point. AI is the great equalizer here — it makes open source platforms dramatically more accessible and manageable than they were even a year ago. The Helm charts, the IAM quirks, the Ranger policies, the Trino catalog configs — all the tedious bits that used to eat weeks now take hours.

- European cloud + open source is genuinely viable now
- AI removes the operational tax that used to kill OSS adoption
- Lakehouse without a vendor — and it still works!

2026-05-02 An agent that writes blogs as me

Personal Saturday Hackathon, and a slightly meta one. I asked Claude to build me an agent that writes blog entries for tigor.nl in my voice. I fed it a handful of my real posts, let it study the cadence, the "Today I..." openings, the occasional bullet list, the candid one-liners, and asked it to produce a system prompt plus a small wrapper that takes a date and a short note from me as input.

The result is surprisingly close. I gave it three test prompts about things I actually did this week, and two out of three I would have published almost as-is. The third was a bit too polished, too LinkedIn-ish — so I tweaked the prompt to explicitly forbid generic conclusions and buzzword summaries. Better.

It's a strange feeling though. The whole point of this blog was to document my learning journey in my words. Now I'm outsourcing the words themselves. I think I'm okay with it as long as the input — the actual hands-on work — stays mine. The agent is a scribe, not the engineer.

- Few-shot examples beat long style instructions
- Explicit "do not" rules fix the LinkedIn-tone problem
- This entry is agent-written and posted. So be it.

2025-11-23 Datacamp MLOps course

I've just completed the **MLOps Concepts** course on DataCamp, marking an exciting step in my journey toward mastering machine learning operations. This course gave me a solid understanding of how to bridge the gap between machine learning development and production deployment. I learned about key principles like model versioning, CI/CD pipelines for ML, monitoring, and automation in the ML lifecycle. It also highlighted the importance of collaboration between data scientists, engineers, and DevOps teams for scalable AI solutions. I'm excited to apply these concepts to real-world projects and continue exploring advanced MLOps tools and best practices.

2025-10-25 Databricks, MLOps, MLFlow etc

Over the past few weeks, I've been digging deep into **how data engineers can better support data scientists and BI teams** — not just by moving data around, but by building the right *architecture* and *workflow* around it.
What I discovered is that **modern MLOps** is really about unifying data engineering, machine learning, and analytics under one governed, automated platform.

Here's what stood out to me:

-

The **Feature Store** (Databricks Feature Store or Feast) is a game-changer. It ensures that features used in model training are exactly the same ones used in production — no more hidden discrepancies or duplicated logic.

-

MLflow ties everything together with experiment tracking and a model registry. I loved how it keeps every training run traceable: parameters, metrics, and artifacts all versioned and reproducible.

-

Model Serving through **Databricks Model Serving** or **Azure ML Endpoints** makes deployment almost seamless — turning a model into an API with CI/CD gates felt surprisingly elegant.

-

Finally, **monitoring and data quality** (using Great Expectations and Evidently) closes the loop. It's not enough to deploy a model; you have to keep an eye on data drift, prediction drift, and overall model health.

When all these components live inside a **Delta Lakehouse** governed by **Unity Catalog** and automated through
Azure DevOps
, you get a true end-to-end ecosystem: reproducible, observable, and scalable — for both BI and ML.

For me, this exploration changed how I think about data engineering. It's no longer just about pipelines and storage — it's about **building the foundation that lets machine learning and analytics thrive together** .

2025-08-13 Fabric AI Foundry

- Still working on API access in Fabric, needed help from Microsoft. Worked a bit on streamlining KeyVault usage and overall maintainability of the python notebook.

The future of AI is here with Microsoft's AI Foundry, Fabric, and Copilot Studio!
With AI Foundry, you can design powerful, intelligent agents that reason, plan, and act. Fabric brings your enterprise data to life—turning natural questions into insights with the Fabric Data Agent. And with
Copilot Studio
, anyone can build custom copilots that connect to Fabric and other tools, no coding required! Together, they form a game-changing ecosystem where
data, intelligence, and user-friendly copilots unite

to supercharge productivity.

2025-08-09 Something else: The Hidden Cost of the Wet DBA

The Wet DBA may have been introduced with good intentions, but its real-world effects are often harmful for nearly everyone involved—except intermediaries.

1.

Bad for companies – Costs increase as they pay higher rates to cover extra compliance, risk, and intermediary margins. Hiring becomes slower and more complex, with legal uncertainty discouraging flexible staffing.

2.

Bad for freelancers (zzp'ers) – Honest competition is undermined. Skilled professionals are passed over for less qualified but "safer" hires. Assignments shrink in number, especially for independent specialists, leading to lost income and wasted potential.

3.

Only good for intermediaries – They can position themselves as a "safe" bridge between companies and freelancers, charging premiums for reducing a legal risk that shouldn't exist in the first place.

In the higher tariff range, the result is a costly policy misstep—one that drains resources from businesses, reduces opportunities for freelancers, and delivers no net benefit to society. All of this stems from regulations shaped more by theory than by practical understanding of the labour market.

2025-08-02 Fabric REST-API for promotion

While using the Microsoft Fabric REST API, I ran into a few setup steps that weren't immediately obvious. I had to:

-

Create an **App Registration** in Azure AD

-

Set the right **API permissions** and create a **client secret**

-

Assign the **service principal** access at the **workspace level** in Fabric

-

Ensure the ADLS Gen2 storage had the correct **role assignments**

Once that was all configured, the integration worked as expected — but it took some digging to align access across all layers.

2025-07-26 Moving to Fabric F2 capacity

Transitioning from the Microsoft Fabric trial to a full paid capacity is simple and ensures your projects continue without interruption. First, purchase a Fabric capacity (like F2 or F4) in the Azure Portal. Then, assign your existing workspaces to this new capacity via the Power BI Admin Portal. All your data and artifacts — like lakehouses, reports, and pipelines — remain intact. This move unlocks production-grade scalability, dedicated resources, and full control over performance and cost. Don forget to pause :-)

2025-07-19 Structuring Microsoft Fabric for CI/CD with GitHub Flow

A best-practice setup in Microsoft Fabric starts with separate workspaces for each stage: Development, (optional) Test, Acceptance, and Production. Each workspace is connected to a different GitHub branch — for example, feature/* for DEV and main for PROD — enabling clean version control and promotion via pull requests. With Git integration, changes can be committed directly from the Fabric UI and deployed through GitHub Flow. Organizing your repo with clear folders for notebooks, pipelines, and models, and automating deployments with GitHub Actions, ensures scalable, secure, and trackable data development across your OTAP pipeline.

2025-07-12 Fabric and GIT Integration

Microsoft Fabric is steadily becoming a more developer-friendly platform — and Git integration for notebooks is a great example of that. Once Git is connected to your workspace, versioning and collaborating on notebooks becomes much more manageable.After the initial setup, the process is smooth. You get a clear view of which objects (like notebooks, reports, and dataflows) have been changed. Everything is shown in the **Source Control** panel, where you can add a commit message and push your changes directly to your Git repository.This brings familiar DevOps practices right into the Fabric experience, making it easier to track changes, collaborate in teams, and maintain clean development workflows. No more downloading and uploading files manually — version control is just part of the workspace now.As Git support expands across more Fabric items, this is a great time to start integrating it into your development flow.

2025-07-05 Databrikcs: Automating the creation of the Databricks Asset Bundle

At my current assignment, we use Databricks Asset Bundles to deploy to higher environments with the help of an Azure DevOps pipeline. The Databricks Asset Bundle is created manually.

Today, I explored ways to make this process less manual and more maintainable. Since managing a large number of jobs, clusters, and notebooks by hand is time-consuming, I investigated how to automate the generation of the bundle.yml file. I experimented with using

the Databricks REST API to pull metadata for existing jobs, notebooks, and clusters, then parse that data into a structured YAML template. By doing this, I was able to create a proof-of-concept Python script that lists all jobs, clusters, and notebooks from a Databricks workspace and prints a starter bundle.yml skeleton. This approach could dramatically reduce the manual effort needed to keep the Asset Bundle configuration in sync with what's actually deployed in the workspace.

Promotion of Unity Catalog changes which are not a part of bundel.yml will be addressed in the next post.

2025-06-29 Fabric: Commiting changes to a GIT branch from AzureDevops pipeline

One of the issues I had to find a solution for was cleaning up custom deployment files in Git after deployment is done. It's not a complex task, except the Azure DevOps pipeline has to be able to delete files in a Git branch.

Instead of hardcoding personal access tokens, I configured the Azure DevOps pipeline to use the built-in Project Build Service identity. By granting this identity Contribute permissions on the repository and the target branch, and enabling persistCredentials: true in the YAML checkout step, the pipeline reuses its secure OAuth token to perform git push operations. This avoids handling user secrets and keeps the process fully automated and secure while respecting branch security controls.

To test the behavior, I set a branch policy on main to enforce commits only via pull requests. However, the pipeline was still able to delete files directly from main because the Allow bypass policies
permission was set to
Allow
for the build service identity.

2025-06-28 Fabric: RBAC, Semantic Modeling & Power BI Integration

Today, I configured RBAC (Role-Based Access Control) permissions in Fabric to manage secure, role-specific data access. I built a
semantic model directly on a lakehouse table
, enabling efficient, governed analytics without data duplication. On top of that, I created a Power BI report
to visualize the data and deliver actionable insights.

Fabric continues to improve and really delivers on its promise of a unified, developer-friendly data platform.

2025-06-22 Deployment of Databricks notebooks with AzureDevops pipeline

Today I set up a two-stage Azure DevOps pipeline to automatically deploy a Databricks notebook from Git to my production Tigornl workspace. Connectivity between AzureDevops and KeyVault via service connection, storing SAS token in KeyVault had to be done first. For now I am not using a service principle or managed identity to spare some time. Planning to focus on Fabric anyway.

In the Build stage, the pipeline checks out the Git repo, picks up the notebook, and publishes it as an artifact. In the Deploy stage, it downloads that artifact, connects to the Databricks prod workspace (via secrets from Azure Key Vault), and uploads the notebook using the Databricks CLI.

This setup gives us:

- Version-controlled notebooks
- Reproducible deployments
- Secure separation of environments

No more manual uploads. Just push to Git, and it lands in the right Databricks folder.

2025-06-21 Resursive SQL in Databricks

Recently, I had to write a recursive query to traverse a hierarchy—but ran into a problem. Databricks (using Spark SQL) does not support recursive SQL constructs like Oracle's CONNECT BY PRIOR or SQL Server's recursive CTEs (WITH RECURSIVE). Here's a clear explanation why:

Why recursive SQL isn't supported in Databricks (Spark SQL):

1. Spark SQL is built on distributed computation:
Databricks runs on Apache Spark, which is designed for large-scale distributed data processing. Recursive queries are inherently iterative and stateful, which doesn't map naturally to Spark's parallel, stateless execution model.
2. No built-in optimization for recursion:
Traditional RDBMSs (like Oracle or SQL Server) are row-oriented and execute recursive queries step-by-step with tight control.
Spark, being batch-oriented, would need to repeatedly submit jobs for each iteration. That's expensive and inefficient unless there's tight optimization—something Spark SQL doesn't offer out of the box for recursion.
3. Recursive logic must be handled manually in Spark:
You're expected to use DataFrame APIs with loops or iterative joins programmatically (e.g., in Python or Scala) to mimic recursion.

A straightforward loop-based solution that processed each row individually took 2 days for just 150K rows—not acceptable.

 What did I do instead?

- I translated the datasets I needed from Spark DataFrames to dictionaries to eliminate overhead, since the data was relatively small.
- I wrote a recursive Python function that finds all parents for a given ID. If a parent has a specific property, it gets returned.
- For technical reasons, I converted the output to tuples.
- Then, I applied the function to each row in a Pandas DataFrame. This avoided explicit looping and leveraged internal parallelism.

Result:

Running time dropped from 2 days to just 5 minutes. Huge win!

2025-06-21 Fabric OneLake

Today, I worked with **Microsoft Fabric** and **OneLake**, focusing on integrating **Delta Lake** with **PySpark notebooks**
. I started by creating a sample DataFrame with Flight data, then wrote it to a **Lakehouse**
called Tigornl_lakehouse using the **Delta**
format. I migrated some old Databricks notebook and made it run in Fabric. Some connectivity issues were resolved. Conclusion is: it still work as I would expected. Databricks gives you more control and performance,. However, Fabric will prevail on Azure in a couple of years.

2025-06-14 Fabric!

Today I worked with Microsoft Fabric to build a secure data workflow using notebooks and Azure Key Vault. I started by creating a new Lakehouse and setting up a notebook environment. To access secrets securely, I registered an App in Azure Active Directory and generated a client secret. I then assigned the app the **Key Vault Secrets User** role so it could read secrets from my Key Vault.

In the notebook, I installed the necessary Azure SDK packages using %pip install. I wrote Python code using ClientSecretCredential to authenticate and successfully retrieved a secret from Key Vault. Along the way, I ran into a DefaultAzureCredential error, which taught me more about how authentication works in Fabric's context. I also learned how to pass secrets securely using pipeline parameters instead of hardcoding them.

General thought

Microsoft Fabric will replace Databricks on Azure within 2 to 4 years, much like how SQL Server took market share from Oracle. The key driver is ease of use—Fabric offers a fully unified analytics platform that combines data engineering, warehousing, real-time analytics, and business intelligence in one seamless environment. Unlike Databricks, Fabric is deeply integrated with Microsoft tools like Power BI and Azure Active Directory, simplifying management, security, and governance. Its Lakehouse architecture and OneLake storage provide a single, consistent data source that reduces complexity and silos. Fabric's capacity-based pricing model is more predictable and appealing to enterprises compared to Databricks' usage-based costs. Additionally, Fabric empowers both technical and business users with low-code tools and expanding AI capabilities. For most organizations, Fabric's simplicity and integration will drive it to become the default Azure analytics platform, gradually replacing Databricks in mainstream workloads.

2025-04-19 GitFlow vs GitFHub Flow

I was assisting in DevOps with designing a branching strategy. Having worked at different companies and used various branching strategies, I had never really thought about why someone would use a dev branch and master, or just master, or also include release and separate test branches, etc.

What I've come to realize is this: whenever your process requires more control over complexity and time-consuming coordination, you likely need an extra branch. But if you're able to develop quickly with well-isolated segments of code, I would choose GitHub Flow.

On the other hand, if your process involves extensive testing, multiple development teams, and a high number of merge conflicts, then I'd recommend the less dynamic, slower—but safer—GitFlow.

2025-04-13 AI Assistant vs ChatGPT vs Google

Recently, I was working on a REST API automation task in Databricks. Some of the activities were more related to Platform Engineering, which I used to find a bit boring—until now. This time, I actually got excited!

The concrete problem was that the secret created for a Service Principal in Databricks gave an error about the format of the token when used with the REST API. ChatGPT, the AI assistant in Databricks, and even the official Databricks Docs didn't give me a clear answer.

Luckily, my old friend Google helped! It turned out that the token must be used to fetch another token from the OIDC endpoint first:

response = requests.post(auth=(CLIENT_ID, CLIENT_SECRET), headers = {}

2024-12-08 AI Assistant in Databricks Notebooks

While developing in Databricks, I started using the AI Assistant, and I think it's absolutely amazing. It feels like I'm effectively becoming a "prompt engineer." The boost in productivity is incredible—easily several times greater than before. The Assistant helps with everything from fixing existing code to generating entire scripts based on a functional description. Mastering the skill of asking the right questions has become a key focus, as it's essential to get the best out of this tool.

Of course, you still need to understand the problem and articulate it clearly, but the actual implementation can often be handed over to AI. While I still need to correct and fine-tune some of the generated code, it's usually pretty close to what I need. One of the most impressive features is the Assistant's ability to maintain context based on the active notebook and specific cells, allowing it to anticipate what I might ask next. It feels like a revolution unfolding before my eyes.

At the same time, it's a bit bittersweet. I realize I might be part of the last generation that finds joy in solving technical puzzles and manually writing code from scratch. The landscape of development is changing rapidly, and it's exciting yet nostalgic to witness this transformation.

2024-11-30 Personal Saturday Hackathon: complex XML parsing in Databricks with PySpark

Parsing complex XML structures can be challenging, especially when working with data that includes deeply nested arrays and structs. I recently experimented with PySpark to develop a flexible approach for handling this complexity, allowing me to control what to explode and what to flatten during the parsing process.

When ingesting XML data, the resulting DataFrame typically contains array and struct columns. Exploding all arrays in one pass is not feasible because it leads to a Cartesian product of the arrays, particularly when they have many-to-many relationships at the same level of the hierarchy. To avoid this issue, I focused on selectively exploding specific elements, enabling the generation of multiple tables. These tables can later be connected using unique IDs, preserving the logical relationships within the data. Using explode_outer is critical in this process as it preserves rows that do not contain arrays (hierarchies), ensuring no data is lost.

To further streamline the parsing process, I created a recursive function that handles the complexity of navigating and flattening nested structures. This recursive approach allows for dynamic handling of diverse XML schemas, ensuring a scalable and reusable solution. Notably, I achieved this without relying on XSD schemas, making the solution more adaptable to XML files with undefined or variable structures.

This method is similar to how the SAS DI XML parser operates, as it also breaks down complex XML data into manageable components. By using this controlled approach, I was able to maintain data integrity while efficiently parsing and transforming XML structures.

2024-10-06 Personal Sunday Hackathon: DV automation end to end, an idea and ChatGPT

There are already plenty of automation tools for Data Vault on the market. So why wouldn’t I create another one? (Yes, sarcasm intended! ????) If you can convert a Business Object Model or an ERD (not exactly the same, but close enough for an experiment) into a Data Vault model,

fill in the necessary **AutomateDV** metadata in dbt, and automate the model creation and loading, you have an end-to-end solution. Expensive tools like WhereScape can do this, as far as I understand. The point is, I want to achieve the same without paying a fortune.

The real challenge lies in the first step: converting a Business Object Model into a Data Vault model, which is far from trivial. But as the saying goes, a good developer is a lazy developer. So why wouldn't I use OpenAI's APIs to tackle this? It sounds like a fun challenge to solve! Surprisingly, my first question to ChatGPT—"Can you convert this simple ERD model to a Data Vault model?"—was answered correctly. I must confess, I talk to ChatGPT daily and probably spend more time with it than with any human. It feels like having an intelligent and experienced developer as a colleague, always available. Sure, it occasionally provides inaccurate advice, but overall, it's far better than working alone. My productivity, especially in the areas I'm passionate about, has skyrocketed!

... A few hours later... No, it's even more exciting! I've discovered I can do so much more with AI, and I already have a couple of ideas. Unfortunately, I can't share them just yet.

General Thoughts About AI

When it comes to development as we know it—designing, modeling, coding, testing, and so on—it's clear that AI will eventually handle all these tasks. However, this doesn't mean humans will be replaced. There are two key reasons for this.

Firstly, someone needs to formulate the input and validate the output. This emerging role is called **Prompt Engineering**, and it's becoming a critical skill. Secondly, accountability remains essential. You can't simply say, "Sorry your business was ruined because of our software, but AI is to blame!" Or, "Unfortunately, the patient passed away because AI misdiagnosed." Humans will always need to take responsibility for the outcomes of AI-driven decisions.

2024-09-29 Personal Sunday Hackathon: DBT & AutomateDV package for Data Vault 2.0

From my experience, you need some kind of automation framework in place when working with Data Vault. Without it, you'll end up dealing with a lot of nearly identical and hard-to-maintain code. That's where **dbt** with the **AutomateDV** package becomes incredibly useful.

Although I've worked with more user-friendly (and much more expensive!) visual Data Vault generators, for a developer, dbt combined with AutomateDV is an excellent choice. It offers a more declarative rather than imperative approach to development, which brings structure and, as a result, saves a significant amount of time and effort.

AutomateDV supports the **Data Vault 2.0 standards** and provides a full-featured, insert-only Data Vault model and loading implementation tool. It includes macros for key components such as hash keys, hash diffs, PIT tables, Multi-Active Satellites, Bridge tables, Effective Period Satellites, Reference tables, and more. Additionally, it supports automated testing and documentation, which is, of course, awesome.

We all know how testing and documentation often get pushed to the very end of a sprint or project and are the first things to be sacrificed when deadlines become tight. AutomateDV addresses this problem by integrating these components into the development process, ensuring they aren't overlooked.

2024-09-15 Personal Sunday Hackathon: DBT & Databricks

DBT in the context of Databricks has been gaining a lot of attention recently. As the "T" in Extract, Load, Transform (ELT), dbt makes Databricks more accessible for Data Engineering teams that are SQL-minded. Meanwhile, Data Scientists, who are often accustomed to using Python and PySpark notebooks for machine learning, find Databricks an easy choice when companies seek a unified platform for their data needs.

However, connecting dbt to a Unity Catalog in Databricks wasn't entirely straightforward. It required setting inbound firewall rules on the Security Group, which were overridden by Deny Assignments due to Databricks' adherence to Best Practices. Fortunately, Microsoft Support proved invaluable in resolving these issues and ensuring a smooth setup experience. So far, their assistance has been excellent! ????

2024-09-01 Personal Sunday Hackaton: RESTfull API's in Databricks

Added a couple of Databricks Notebooks to ingest CBS data using OData RESTFul API's. Some useful information about OData and REST (sorry, it is mainly for myself :-))

ink

- REST is an architectural style for exchanging information via the HTTP protocol, while OData builds on top of REST to define best practices for building REST APIs.
- Both REST and OData adhere to web-based architecture, statelessness, data format flexibility, resource-oriented design, and interoperability.
- OData offers enhanced query capabilities and standardized metadata, promoting discoverability and efficient data retrieval.
- REST provides simplicity, flexibility, wide adoption, and platform independence, making it suitable for a broad range of applications.
- Considerations for choosing between OData and REST include complexity, interoperability, and alignment with project requirements.

2024-07-14 Personal Saturday Hackathon: A dive into Fabric... or not...

Figuring out how everything works within Fabric was quite time-consuming, but in the end, it all made sense. Understanding how the different capacities—Power BI Pro, Premium, and Fabric—relate to their respective capabilities and limitations was also a bit of a challenge but an interesting one to work through.

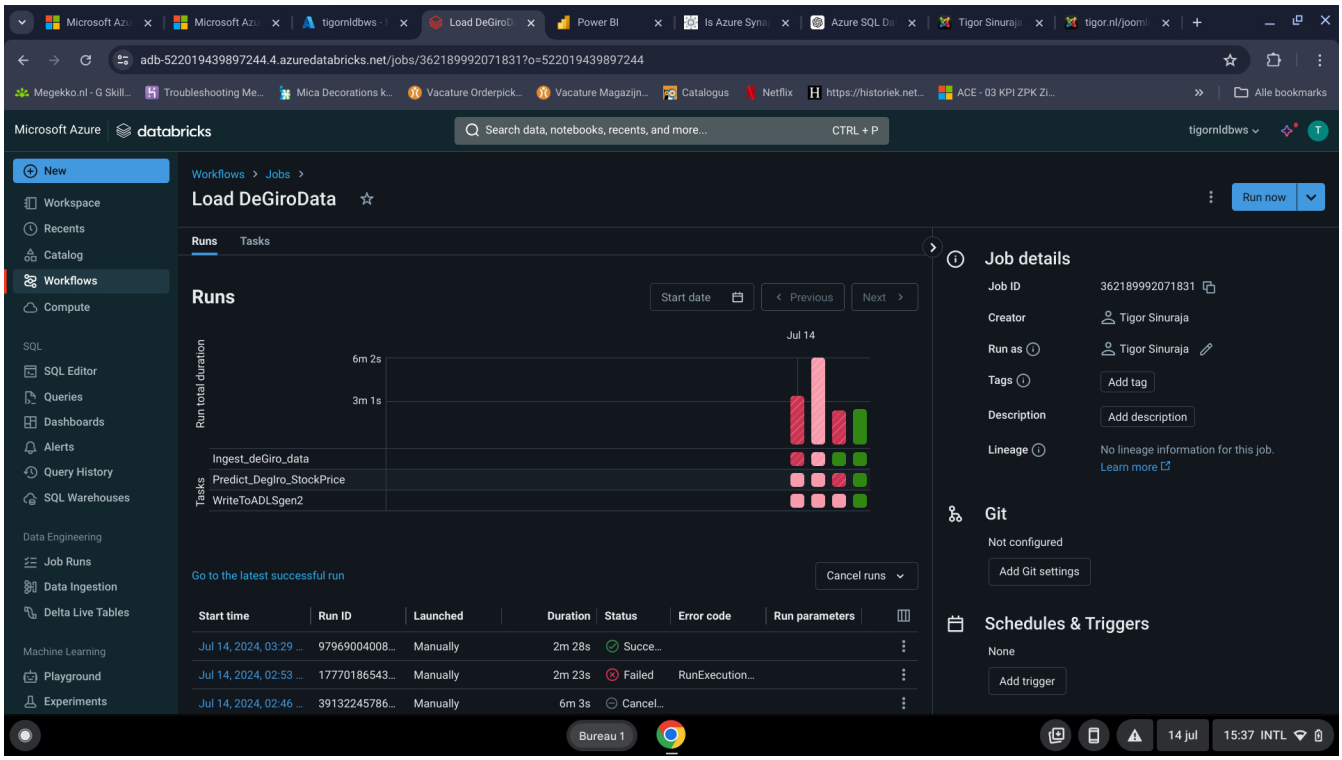
By chance, I came across an article that confirmed my previous experiences with Microsoft’s Cloud Data Platform. It was reassuring to see my observations validated and added more context to what I had already learned.

Link

As mentioned earlier, Databricks plays a key role in the workflow where data from DeGiro is ingested, the model is trained, and predictions are generated. These actions are part of a

scheduled **Databricks Workflow** that runs daily. While the functionality includes support for streaming APIs, this feature is currently disabled.

For now, the workflow integrates three notebooks: **Batch ingest DeGiro data**, **Train model**, and
Predict values and write to ADLS2
. Throughout the process, all intermediate results—such as the training set, prediction set, and final output—are written to
Delta format tables
, ensuring high performance, reliability, and scalability for data storage and processing.



What interesting is to see how predictons compare to real values. Although it has never been the goal (i was primarily busy with technologies) it is good to see that it al least shows some similar trend as reality (red line)

I created a dynamic version of the test report. Don’t mind the content—it’s functionally meaningless. The objective here was to explore the technology. I wondered if it could be done without using Power BI Desktop. To my surprise, the only available data source options were CSV, XLS, or a Published Semantic Model. The message was clear: *"If more is needed, please use Power BI Desktop."*
Right... And since I didn’t want to use Fabric capacity (to avoid extra costs), I followed these steps:

- Previously, I created Databricks notebooks to ingest data, train an ML model, and write the results to a mounted ADLS location. Within Synapse Analytics, I defined external tables in a SQL database that point to this ADLS location. I considered using a Synapse Lake Database but decided against it since it requires managing a Spark pool, which adds complexity.
- Next, I created a **Dataflow** in Power BI. The Dataflow doesn’t perform any transformations—it simply connects to Synapse Analytics. Initially, I thought I would need a pipeline, but thankfully, it wasn’t required. During this process, a Semantic Model was created. (Correction: the Semantic Model was actually created when I connected to the ADLS2 location, not to the external tables in Synapse Analytics.)
- Unfortunately, it seems there is no way to connect without using Power BI Desktop or Analysis Services. That Semantic Model was then used to create the report, which I subsequently published to the website.

2024-06-22 About certifications

In today's rapidly evolving technological landscape, it has become increasingly challenging to decide which tools and technologies warrant the investment of time and resources. Achieving proficiency in any skill demands significant effort, and with the swift pace of innovation, the longevity of such expertise is often uncertain.

Take, for instance, Microsoft's Azure Synapse Analytics. Introduced as an evolution of SQL Data Warehouse, Synapse has been a cornerstone for data professionals. However, with the advent of Microsoft Fabric, there's a noticeable shift in focus. While Microsoft has not officially announced the deprecation of Synapse Analytics, the introduction of Fabric suggests a strategic move towards a more integrated data platform.

This scenario underscores a broader challenge: investing time and effort into mastering specific tools can be risky if those tools may become obsolete or significantly altered in a short span. While core principles and foundational knowledge remain valuable, the specifics of version-dependent features might lose relevance quickly.

2024-06-22 Personal Saturday Hackathon: Databricks, Fabric, PowerBI

Published to web PowerBI report powered by Fabric. Databricks notebooks are used to extract data from DeGiro, train a Machine Learning model and predict the value of Roblox share (that one is my son's favorite). The result is written to an ADLS mount. Form there it is picked up by PowerBI report created in Fabric and refreshed daily. Except some connectivity and authorization plumbing it all went smoothly. Beneath is not a screenshot, it is a PowerBI report :-)

Live PowerBI report

2024-04-19 Databricks LakeHouse, Delta Tables and Medallion architecture

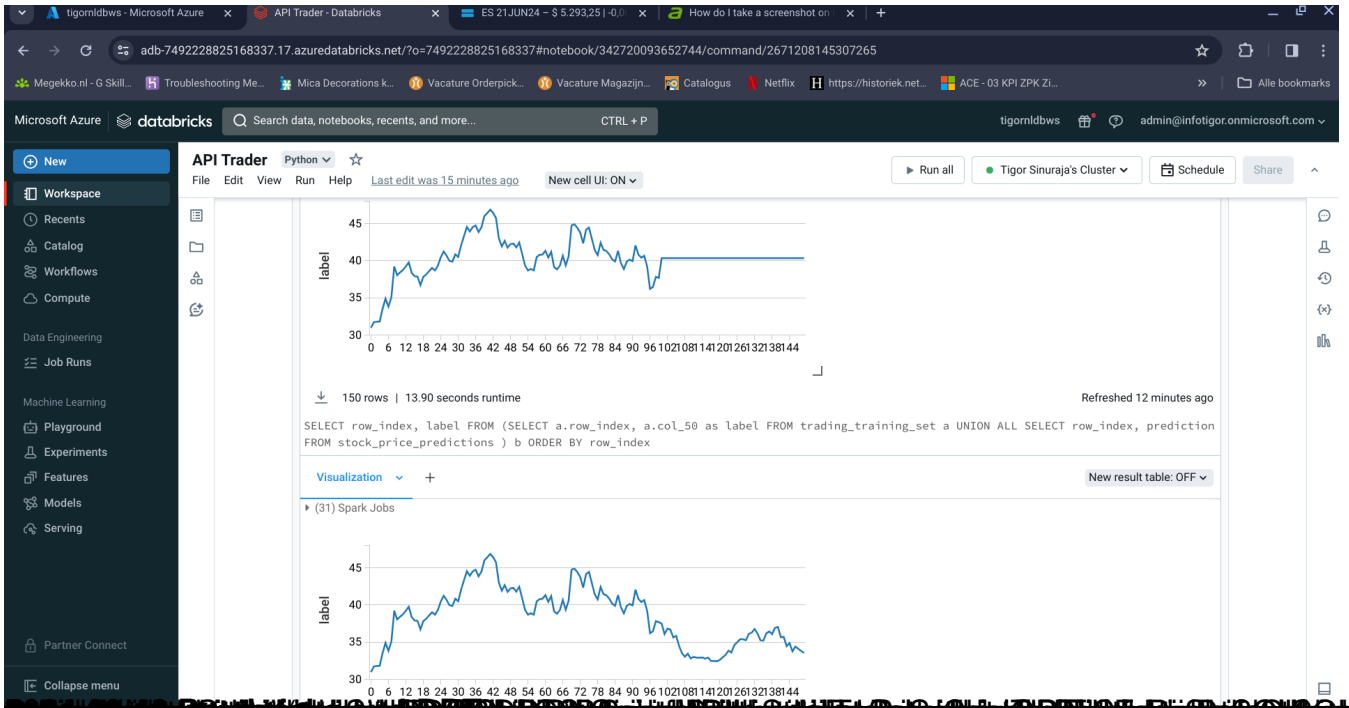
Just like the whole data world is transitioning form traditional Datawarehousing to Data Lakes and to LakeHouses I am moving too. Frankly speaking, otherwise it would be simply boring! So that is always exciting, it gives you energy. On the other hand, you have to learn non stop and in ever increasing pace.

I am currently working on an on-premise Data Lake. So the logical step forward is to combine that knowledge with the years of experience with traditional Datawarehouses and start building a Lake House. I am working already on an idea to use Tigor.nl which I own for decades as a platform for it.

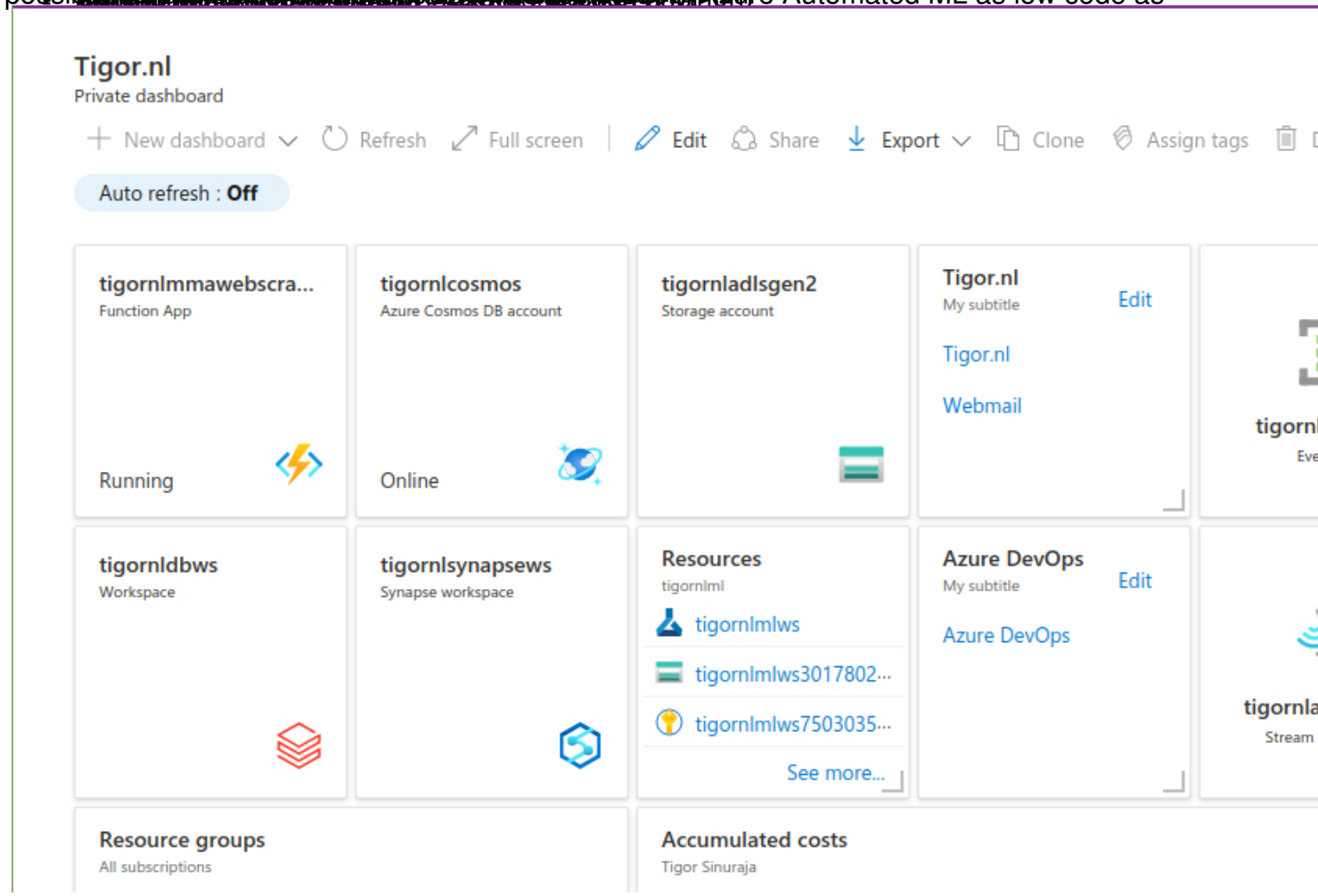
2024-03-23 Saturday Hackathon: streaming data, Delta Tables, Machine Learning n Databricks
notebooks found from 2 years ago.

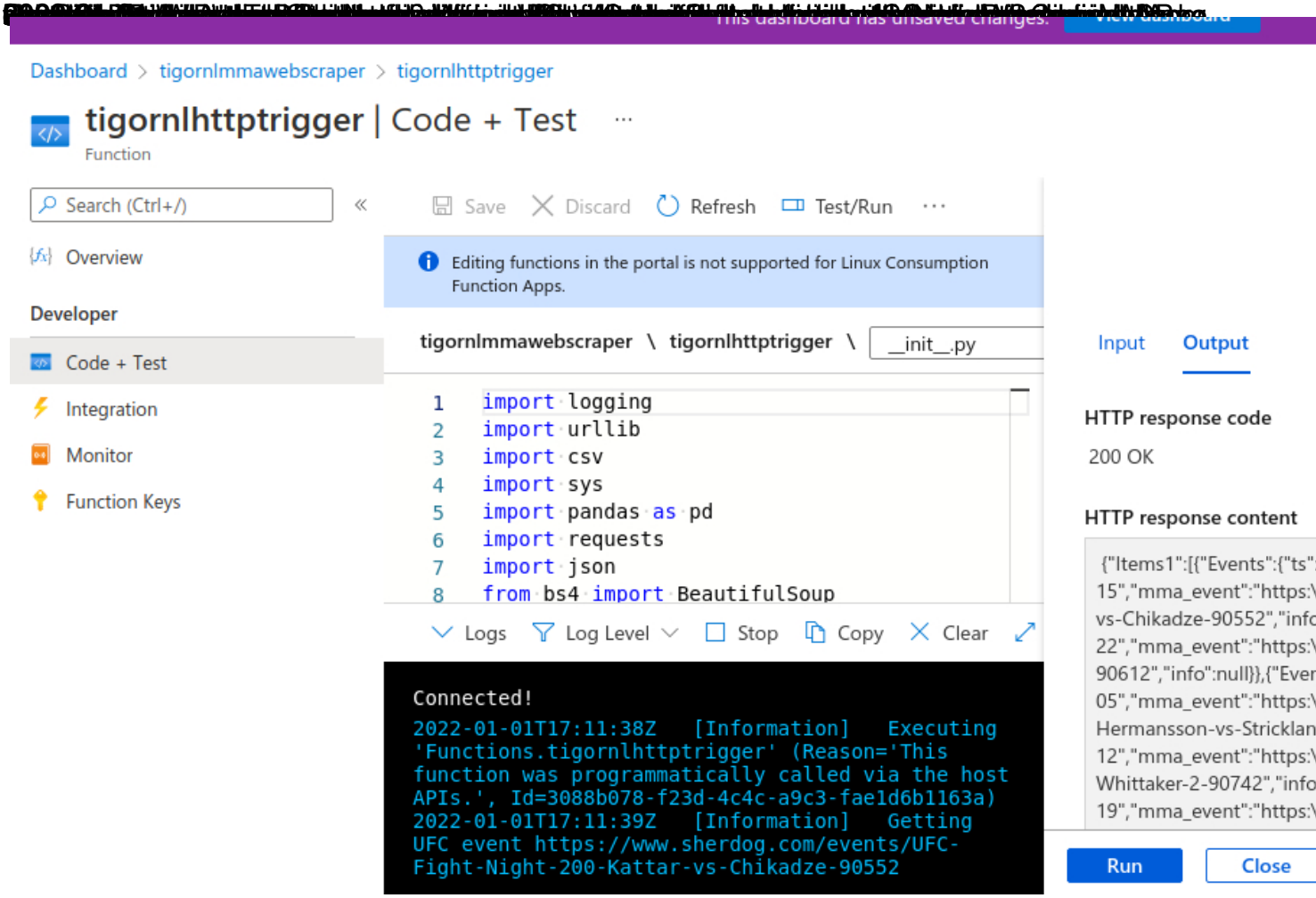
A couple of Saturdays were dedicated to get streaming system going using Structured Streaming APIs with FlightRadar24 data in Databricks. In that proces I found notebooks I created two years ago. Actually, I was surprised by things I could do :-). Using degiroapi get data from the DeGiro site, the stock price of Roblox stock and using pyspark and Random Forest model I predicted the stock price. And it still works! although I had to hack degiroapi a bit.

Oh yeah, and I bought Microsoft Support for my subscription. My first experience was actually great. I was immedately helped by 2 support people and afterwards I was interviewed by their manager and he asked if I was satisfied. Yes, I was!



2022-[redacted] possi[redacted] Automated ML as low code as




This dashboard has unsaved changes.
Dashboard > tigornlmmawebscraper > tigornlhttptrigger
tigornlhttptrigger | Code + Test
Function
Search (Ctrl+/)
Overview
Developer
Code + Test
Integration
Monitor
Function Keys
Save
Discard
Refresh
Test/Run
Editing functions in the portal is not supported for Linux Consumption Function Apps.
tigornlmmawebscraper \ tigornlhttptrigger \ __init__.py
1 import logging
2 import urllib
3 import csv
4 import sys
5 import pandas as pd
6 import requests
7 import json
8 from bs4 import BeautifulSoup
Logs
Log Level
Stop
Copy
Clear
Connected!
2022-01-01T17:11:38Z [Information] Executing 'Functions.tigornlhttptrigger' (Reason='This function was programmatically called via the host APIs.', Id=3088b078-f23d-4c4c-a9c3-fae1d6b1163a)
2022-01-01T17:11:39Z [Information] Getting UFC event https://www.sherdog.com/events/UFC-Fight-Night-200-Kattar-vs-Chikadze-90552
Input
Output
HTTP response code
200 OK
HTTP response content
{"Items1":[{"Events":{"ts"
15","mma_event":"https:\
vs-Chikadze-90552","info
22","mma_event":"https:\
90612","info":null}},{"Ever
05","mma_event":"https:\
Hermansson-vs-Stricklan
12","mma_event":"https:\
Whittaker-2-90742","info
19","mma_event":"https:\
Run
Close