

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.

# 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 TigornI 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. Datab ricks (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 Al

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 :-))

#### <u>ink</u>

- 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.

# <u>Link</u>

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!



# 2024-03-10 Databricks and DBT

Lately I have been looking at Databricks again. With my recent experiences with PySpark I want to use it in a modern scalable and easy to use framework. To my surprise Databaricks has SQL datawarehouse en integraion with DBT. Basically they want to make it easy who know SQL (which is basically everyone) to use Databricks which totally makes sense. The only thing which is funny about it is that you dont need PysPark at all, at least if you are a Data Engineer. OKay. But I want something else than SQL. Otherwise, it is just like Snowflake, which is great by the way. But I want to play with PySPark :-). Somehow, being busy with no Low Code developement in Visual Studio Code and Azure Data Studio makes me feel like developer again. And thats a good feeling. Whith my recent interactions with ChatGPT it also is a lot of fun. I guess you have to be Data Scientist to use Python and PySpark. Thats where Databricks is superior to Snowflake and Synapse. So be it. I am going to go ahead and devote my Saturday Hackaton (probably more than one) to making a system which takes a streaming data and predicts something.

To be continued...

#### 2023-04-06 What if ?

#### (unsolicited advice of an old man to those who decide)

What if I was a Head BI and I have to choose an architecture, BI-platform and tooling ? Green field, so to speak. What would be the winning combination ?

Having worked with a number of Dutch companies. Having hands-on experience with software and technologies like SAS, Microsoft Azure, Snowfake, Oracle, Hadoop, Spark, Informatica etc. Having done that on-prem and in the Cloud. Having build datawarehouses on relational databases and Data Lakes. Having modeled with Data Vault and without it.

it's not an easy choice to make!

#### Goal

If I have to sum up what I (as a hypothetical Head of BI) would have wanted to achieve:

- Maximum output: you want to be able to have tangible results as quick as it is possible

- Minimum cost: that includes time (=money) to learn new skills, actually develop and deliver products

- Maintainability: you can replace people easily and you don't have to keep them because they know how one app works

- Continuity: make sure whatever you choose and buy, you don"t have to reconsider within one year

#### Challenges

Why is that so difficult?

- There is no Silver Bullet. There is no single solution for all. But there is probably objectively speaking one optimal choice for your specific situation.

- Datawarehouse, Data Lake or something else? That's an important one. Will talk about it in a separate chapter.

- The number of tools is growing. New vendors coming up continuously, little real experience with those tools is available, but they promise wonders

- Tools are evaluating. What seemed like an optimal choice yesterday can be not so obvious tomorrow

- Information about those tools is sometimes contradictory and depends on where it comes from. Benchmarks are often biased

# Basic principles

Of course, if you have to make a decision of this magnitude, it has to be based on a clear vision and principles and if possible, real life facts and experience

- Hire and keep a good architect :-) Errors in design are very expensive to correct if the system is already build

- Keep it simple when it comes to structuring and modeling your Datawarehouse or Data Lake. Do you understand what a Data Vault is, what problem it solves en do you have those problems at all?

- Keep it simple when organizing development process. Being developer myself for 20+ years I love making things. I love developing, low code or not, making code generators that make code. It is all good until someone else has to take over. For them it's often much less fun. Besides, your company suddenly confronted with the fact that it created critical software which it depends on and it has to maintain and which is not the primary business at all. So avoid it at all cost!

- Make or buy: definitely, if its available, BUY! see previous point.

- No one has canceled basic principles if you have to develop:

- Enforce strict rules and guidelines for development. It does not matter if you do coding or use low code. Otherwise you will get the well known spaghetti and it can be fatal.

- Maintain reusable code in one place. Do not copy code.

- SQL is the only skill here to stay. Organize your development process around it and use it as much as possible. For instance use SparkSQL. (Correction: that one I personally abandoned myself. PySpark is to much fun and power!

- Open Source is only good when you pay for it, and you will, one way or another.
- You don't want to depend on a group of technical people who love what they are doing

(like myself) but who are the only ones who know things. So go for a managed cloud service whenever possible. Yes, Cloud is not cheap, but in the end, it is cheaper! Yes, go Cloud.

to be continued...

**2023-04-06** Unfortunately, Betfair is no longer available in NL putting my MMA -betting project on hold indefinitely. Instead, I am working on an automatic prediction and ordering system based on Degiro. This is done by endless number of people before. What I am aiming for is to use Azure managed cloud services to full extend. I wonder what it would mean in terms of development effort, maintainability and cost,

- Getting historical training data - Azure Function (Python). The main argument to use this technology besides the obvious leaning effect was of course the serverless nature of Azure functions. As the whole point was to leverage the power of Cloud managed services this is a logical choice. My observations so far:

- If you use python, you cant edit in the browser. Developing in Visual Studio Code locally is an excellent alternative though

- To make things work after deployment like importing modules was quite painful even after reading numerous docs and blogs, I would like it to be much easier to handle, It all worked in

the end, but it costed to much time. I want to get a result and I don't care what path I have to set or config I have to fill. It all should be hidden and just work!

- in the case of degiroapi I had to alter request method (otherwise, the site thinks its a bot and refuses connection). Also here it felt like time wasted if it wasn't for the learning effect of it.

- Row data layer - we store collected data in Azure Cosmos DB, JSON files as documents, later I will probably transition to pure Data Lake solution.

- Permanent storage, historization for tranformed data will go to Delta Lake - exciting to use time travel and ACID features

- Data preparation - Synapse, pyspark notebooks, in my opinion, this can be used as alternative to Databricks (for the purpose in mind) and Azure Data Factory. The latter functionality is integrated in Synapse. Dedicated pools I wont use because of the cost. Mainly, its the combination of files in Data Lake as storage and Spark pools for compute.

- Training a model and generate inference model - Azure Automated ML as low code as possible( later may be Databricks and Azure ML SDK)

- Getting instances to predict Azure Function (Python)
- Ad hoc analysis Azure Synapse Serverless pool
- Visualization PowerBI, integrated with Synapse
- Versioning and deployment Azure Devops

**2022-01-02** An idea how the Cloud version of my application is going to be.

- WebScraping of training data Azure Function (Python)
- Store collected data Azure Cosmos DB
- Data preparation Databricks and Azure Data Factory

- Training a model and generate inference model - Azure Automated ML as low code as possible( later may be Databricks and Azure ML SDK)

- Getting instances to predict Azure Function (Python)
- Ad hoc analysis Azure Synapse Serverless pool
- Visualization PowerBI
- Versioning and deployment Azure Devops

#### Written by Administrator Saturday, 01 January 2022 15:59 - Last Updated Saturday, 12 July 2025 06:35

+ New dashboard 🗸 💍	) Refresh 🖉 Full screen	🖉 Edit 👶 Share 🞍 Exp	ort 🗸 🗋 Clone	🖉 Assign	tag
Auto refresh : Off					
tigornImmawebscra Function App	tigornlcosmos Azure Cosmos DB account	tigornladlsgen2 Storage account	Tigor.nl My subtitle Tigor.nl Webmail	Edit	
Running 🥠	Online				
<b>tigornldbws</b> Workspace	tigornlsynapsews Synapse workspace	Resources tigornImI tigornImIws tigornImIws3017802	Azure DevOps My subtitle Azure DevOps	Edit	
	\$	(*) tigornImIws7503035			1

Dashboard > tigornImmawebscraper >	> tigornlhttptrigger	
TigornIhttptrigger	Code + Test	
	🖫 Save 🗙 Discard 🕐 Refresh 🖽 Test/Run \cdots	
{fx} Overview	Editing functions in the portal is not supported for Linux Consumption Function Apps.	
Developer		
Code + Test	tigornImmawebscraper \ tigornIhttptrigger \initpy	Input Output
🗲 Integration	1 import logging 2 import urllib	HTTP response code
Monitor	3 import csv	200 OK
	4 import sys	
Function Keys	5 import pandas as pd	HTTP response content
	6 import requests	•
	7 import json	{"Items1":[{"Events":{"ts"
	<pre>8 from bs4 import BeautifulSoup</pre>	15","mma_event":"https:
	∨ Logs 🍸 Log Level ∨ 🔲 Stop 🗈 Copy 🗙 Clear 🧷	vs-Chikadze-90552","info
	5 5 1 2 15	22","mma_event":"https:"
	Connected	05" "mma_event":"https:/
	2022-01-01T17:11:387 [Information] Executing	Hermansson-vs-Stricklan
	'Functions.tigornlhttptrigger' (Reason='This	12"."mma_event":"https:/
	function was programmatically called via the host	Whittaker-2-90742","info
	APIs.', Id=3088b078-f23d-4c4c-a9c3-faeld6b1163a)	19","mma_event":"https:
	UFC event https://www.sherdog.com/events/UFC-	
	Fight-Night-200-Kattar-vc-Chikadze-00552	Run Close

LIAS IIIISAVE