Field Guide
Deploying AI on Databases

Connect to databases, analyze and visualize with natural language to SQL

15+ Months of Client Operations

1. Lessons
2. Live Apps (4 Variants – 8 Apps)
3. Source Code

GPT-5: Initial Assessment & Live Integration

Live App: app.tigzig.com

DATS-4: Database AI Suite – Version 4

10th August 2025

Deployments

1. **Build by Practitioner. Built for Business**: The DATS-4 suite is built by a data scientist for internal teams. The design prioritizes analytical agility and rapid deployment in secure SMB environments. This involves different trade-offs than the standards for large-scale enterprise software.

2. **Live History**: The first client deployment was in April 2024. There are currently 9 live, customized versions running across 3 SMB clients.

3. **Implementation Variants:** Client projects vary based on need-specific components only, custom GPTs connected to databases, rapid-deploy version & customizations.

4. **The Public App (app.tigzig.com):** Fully functional version of the suite. It has been configured as a minimal security sandbox to allow for unrestricted testing of the core features.

5. **Live Project Checklist**: ALL client projects include a mandatory checklist: security layers, semantic model, fixed database connections, and **disabling of admin features for end users**

DATS-4 Evolution

Database AI Suite – Version 4

	OSS Release	Name	Additional Features
V1	Jun ’24	Analytics Assistant App	▪ Flowise UI + FastAPI for Text-to-SQL ▪ MySQL support ▪ Python charts & stats ▪ ChatGPT connected to Databases
V2	Nov’24	REX-2	▪ React UI ▪ Flowise chatflow backend ▪ Postgres support ▪ Interactive grid ▪ Direct file upload to DB ▪ PDF reports ▪ Quick analysis options ▪ OAuth
V3	Feb’25	REX-3	▪ Multi-step reasoning based analysis ▪ Choice of multiple LLM ▪ Flowise sequential agent backend ▪ Agent reasoning view ▪ Quick try sample functionality ▪ Logs
V4	Aug’25	DATS -4	▪ Flowise new multi agent backend ▪ Updated LLM Choices w/ GPT-5 ▪ Database table export & CSV Download ▪ Export to PDF (Text only) ▪ Updated UI ▪ Portfolio analyst integration

Field Report: GPT-5 First Look

From my experience, new model releases often have higher, volatile latencies and costs in the first days or weeks, then stabilize over time.

My preliminary assessment of GPT-5 first few days of release as of 10th Aug 2025:

1. **Reasoning & Analysis**: close to Claude Sonnet 4
2. **Latencies:** higher - temporary phenomenon
3. **Costs:** higher than expected. Likely temporary. On watch.
4. **Variance:** Given the amount of variance I am seeing with GPT-5, cost estimates would not be reliable. I am holding off on sharing exact cost estimates for GPT-5 for the time being. But based on published rates, I expect them to stabilize around GPT-4.1 levels
5. **Integration:** At the same time, I have incorporated GPT-5 as an LLM Choice options in the public DATS-4 for users to try out and compare results
6. **Detailed cost comparisons** and model choices are covered in the LLM section later in this guide

---

I typically migrate clients only once I am confident that the model performance, cost and latencies have stabilized.

---

1. Lessons

1. Security
2. Datamart & Context
3. Agent Setups
4. LLM Choices
5. LLM Cost
6. Usage Patterns
7. Platforms

Security

1. **Align with rules** – set by DB and server admins. They are troublesome but **will save your bacon one day.**
2. **No end user touches the raw tables**– even with SELECT access
3. **Separate user ID's** at DB level with fine grained permissions
4. **Row Level Security** – use with Postgres
5. **Separate** : schemas / database / views for say Finance vs. Marketing. The additional maintenance effort is worth it.
6. **Authentication**: OAuth / API Keys
7. **Log all API calls :** push to a DB / 3rd party tools
8. **IP / Domain Whitelist** : FastAPI / DBs / Agents / all end-points
9. **CORS** : for all FastAPI, with domain whitelist
10. **Resource Limits for CPU & Memory** – implement on server
11. **Rate Limits**: at FastAPI (with SlowAPI) and Agent end
12. **Server**: Firewalls, only SSH, Fail2Ban, IP Whitelists etc.
13. **VPNs**: default deployment always on a client VPN.

Security is expensive – direct cost, bandwidth and business opportunity loss. Every layer adds cost and user friction. Assess risk of breach for each data item, worse cases and potential impact. Apply layers accordingly. Everything is not catastrophic.

Building the Foundation
Datamarts and AI Context

Datamarts

Creating usable datamarts is one of the most time consuming things, especially the **data cleaning and validating** against reported numbers.

1. **Need to know:** create custom datamarts and views for specific use cases. Operate on need to know basis.
2. **Auto Refresh**: setup auto refresh of datamarts
3. **Validation reports**: validation reports for all datamart refreshes is mandatory
4. **Be alert: After running for months, a validation can suddenly fail out of the blue. You must be ready to catch it.**

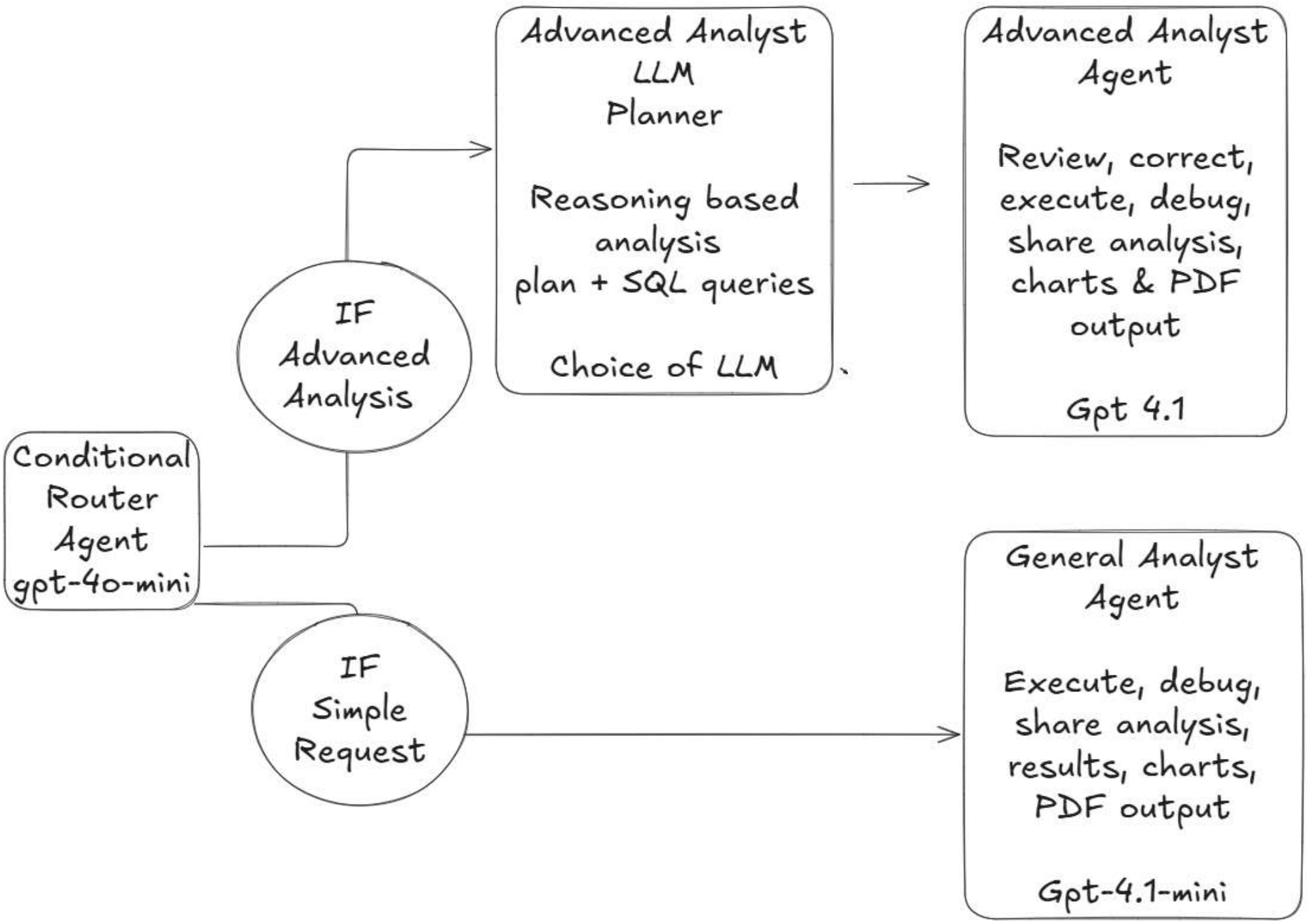
Context

Use system prompt to provide context to AI

1. Sample rows
2. Univariates for numerics & distributions for categoricals
3. Business rules and business context
4. Golden queries – sample queries for common requests, particularly for the more complex queries
5. Output formats / row limits / data gotchas

Agent Setup

Agent backend on Flowise AI, with conditional routing based on type of request. A multi-sequential agent setup



Conditional routing agent will route to advanced analyst or general analyst based on a set of guidelines or if specifically instructed by user

Agent Setup

1. **The Dispatcher (Conditional Router Agent)**

This is the gatekeeper. Its only job is to analyze the user's request and route it to correct specialist agent based on a set of rules.

2. **The Workhorse (General Analyst Agent)**

This is GPT-4.1 mini - optimized for execution speed. It handles the majority of requests: direct SQL queries, data pulls, and standard charts. It does not perform multi-step reasoning. It directly executes, validates, and returns the result.

3. **The Specialist (Advanced Analyst)**

This is a two-step routing, used only for complex requests that require reasoning.

A. The Planner: First, a reasoning-focused LLM (choice of LLMs) creates a step-by-step analysis plan, including the exact SQL and python code required.

B. The Executor Agent: This is GPT-4.1 in all cases - reviews and executes that plan, performing final error checks and formatting the output.

The Executor agent will be upgraded to GPT-5 series once its cost and latency have stabilized

Equipping the Agents: Core Tools

The core DATS-4 agent uses three primary tools:

1. **Database Connect**
   - Custom FastAPI Server
   - Allows agent to connect to database to execute SQL queries
2. **e2b Code Interpreter**
   - Flowise built-in tool
   - Python sandbox
   - To create charts and run statistical analysis
3. **Markdown to PDF**
   - MCP Server
   - To create PDF (text only) output. The agent sends markdown to the MCP Server, which returns a PDF file path.

The system is modular, allowing other tools to be plugged in as needed: web scrapers, Excel updaters, report emailers, file converters, custom automations and more.

Agent Orchestration

To get an agent to deliver the right outcome, you have to test and calibrate- sometimes 100s of times. It's the only way. The are the rules I follow

1. **No 100%** : You will never get 100% what you instruct 100% of the time. Test and determine what variance you can live with.
2. **Edge cases**: test edge cases and outliers. Calibrate instruction till you get your desired outcome
3. **Break it** : Push it to limits. See where it trips and falls.
4. **Reasoning required ?** – if so , specify . Not always required.
5. **Number of Queries** – CRITICAL to specify a cap on number of SQL queries an agent can run for a single question.
6. **CREATE / ALTER/ DROP** : specify if they are allowed or not
7. **Temporary tables** : specify if permitted and how (CREATE TEMP or CREATE TABLE) , and cleanup protocols
8. **Limit clause**: how many rows ? When to use ? When not?
9. **Division by zero**: common error – COALESCE(), NULLIF() etc
10. **Debug** : debugging protocol for query failures
11. **Reminders help** – remind to check for common issues – missing table, table exists, joins, data type mismatches etc

Agent Backend

Don’t reinvent the wheel.

1. **Don’t reinvent the wheel:** use tools like Flowise/ n8n as first choice- they take care of many nuances out-of-box. Connect user interface via API calls.

2. **Flowise AI**: **is my first choice**. Robust out-of-box memory and state management and numerous other features. Great for complex agent workflow, especially for sequential flows.

3. **n8n** – for app integrations and where Flowise not the best fit.

4. **Hard-coded agents**: used only for functionality that framework tools can't support.

LLM Choices

For end-user applications, use frontline providers (OpenAI, Google, Anthropic). They offer the best combination of reliability, consistency, quality and pricing. For internal analytics work – practitioners should test and use other models per their own judgment.

My Top Recommendations

1. **SQL Executions**: GPT-4.1 (GPT-5 once stable) for complex and 4.1-mini (GPT-5-mini once stable) for rest. GPT-4o-mini is excellent for simpler and repetitive requests.
2. **Tool use:** for all tool use functions, OpenAI's GPT models - effective, reliable and cost efficient
3. **Non-Tool LLM use**: Gemini Flash 2.0/ 2.5 as first choice for non tool tasks - e.g. automations, schema detection, reasoning, planning
4. **Complex**: Claude Sonnet 4 for the hardest and most complex tasks
5. **Other LLMs:** DATS-4 provides LLM choices including DeepSeek, Qwen & GLM. Great quality and pricing. But I see a lot of variance in billed cost and latencies. DATS-4 allows for easy integration of other LLMs

LLM Costs: Guidelines

- **Use Case**: Always estimate for your specific use case. Check actual charged API costs. Don't rely on published rate.
- **Lowest Cost**: GPT-4o mini and Gemini Flash 2.0 are older model, but robust, lowest cost and great for many tasks. Test them first
- **Value**: GPT-4.1-mini (GPT-5-mini once stable) and Gemini Flash 2.5 - great workhorses at reasonable cost.
- **GPT-4.1** for harder tasks especially complex SQL executions (GPT-5 once stable)
- **Claude Sonnet 4.0 is an all rounder** and the best, but expensive. Keep for most complex reasoning.
- **DeepSeek. Qwen, GLM and others** – High latency and cost variance based on provider. DeepSeek more stable now.
- **Single step** agents for direct questions = low cost.
- **Multi-step** agents = exponential cost increase. Use with care. See next sections
- **Number of SQL queries** an agent is allowed has direct cost impact. 2 queries per question vs. 10 queries= 5x cost
- **Context** – piles up with same session adding to cost. New question = open new session

Cost estimates as of 10th August 2025. Verify current rates before budgeting.

LLM Costs: 1 Question 1 Query

Cost Per 100 Simple Questions

1 Question = 1 SQL Queries/ Tool Call

Single step: no reasoning step, direct execution

~USD per 100 Q

LLM	**~USD**	**Remarks**
GPT-5	Volatile	New release - high variance. Expect to stabilize around GPT 4.1 levels
GPT-4.1	2.0	Best for complex SQLs
GPT-4.1-mini	0.50	Great for med. complexity SQL
GPT-4o- mini	0.25	Great for simple/ med. complex

Example of single question

1. *Share sample rows*
2. *Add new columns as per instructions*
3. *Join Table A & B by cust_id*
4. *Summarize by housing and show counts*
5. *Share chart for housing summary*

- Actual vary by use case and the agent setup. Always estimate for your use cases and compare vs. actuals.
- For all estimates – keep in mind that as context increases the cost goes higher

Cost estimates as of 10th August 2025. Verify current rates before budgeting.

LLM Costs : Advanced Analysis

- In multi-step reasoning-based analysis, execution cost is biggest chunk due to multiple tool calls.
- 1 Question = 7-10 SQL Queries / Tool Calls.
- All executions by GPT-4.1. GPT-5 costs, once stable, likely to be around same levels.
- Time: ~2-3 mins per que. Can go upto 10m. Varies by question.

~USD per 100 Q

Reasoning Model	Quality Score	Logic USD	Exec. USD	Total USD	Remarks
Gemini Flash 2.0	75	0.25	12.5	12.75	Best value
Gemini Flash 2.5	75	1.75	12.5	14.25	Next after Flash 2.0
Gemini Pro 2.5	85	8.50	12.5	21.00	Avoid. V.High.Cost
Claude Sonnet 4	100	6.50	12.5	19.00	Topmost Quality
DeepSeek R1	90	2.25	12.5	14.75	Great Value
Qwen 3	75	3.50	12.5	13.25	High variances
GLM 4.5	80	1.00	12.5	13.50	High variances
o4-Mini	75	2.75	12.5	15.25	Avoid.
GPT-4.1	90	3.00	12.5	15.50	Great Value
GPT-5	95	Volatile			Top Quality

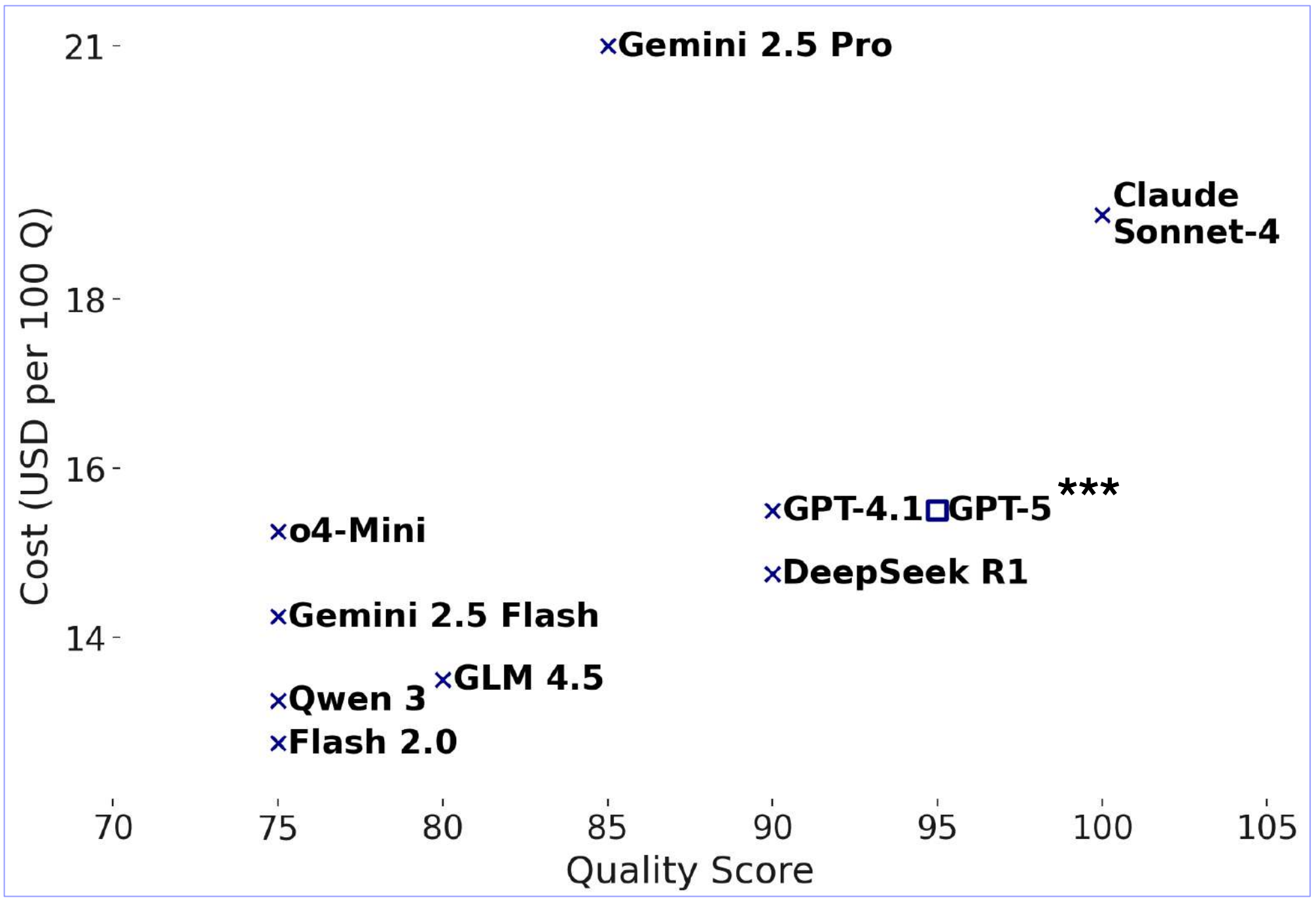
Example of one Advanced Analysis Question (shortened)

1. Create Weighted Average score based on available variables
2. Modelling Data Mart : Take transaction table, summarize based on cust_id and create derived variables. Summarize and merge with customer data to create a modelling data mart

Cost estimates as of 10th August 2025. Verify current rates before budgeting.

Advanced Analysis: Costs Vs. Quality Matrix

Claude Sonnet-4 and GPT-5 are top-tier models for advanced reasoning.



- Estimates based on live deployments & 250+ test runs
- Quality scores are a judgment-based assessment of the model's analytical reasoning depth
- Estimates vary – always estimate and check actuals for your use cases

**** GPT-5 is plotted at its **projected** stabilized cost (equal to GPT-4.1) for quality comparison only. Current costs are volatile and are not plotted..*

Cost estimates as of 10th August 2025. Verify current rates before budgeting.

Warning: The Cost Multipliers of Multi-Step Agents

For multi-step analysis, costs don't just go up; they escalate exponentially - from 10X to 50X or more. This is a critical budget risk. Key factors:

1. **No. of Steps:**
   2 Step = double the context = 2X the cost
2. **Number of tool calls**
   Determined by # of SQL Queries allowed
   Simple Question = 1 SQL Query
   Analysis Question = 2 to 10+ SQL Queries
   =10X cost
3. **Execution Model**
   Needs stronger model GPT 4.1 vs 4.1 mini = 5X
4. **Additional LLM Cost for reasoning-**
   ~ 7 cents per question for Sonnet 4
5. **Context is a Multiplier**, not an addition: larger semantic models, context, and system instructions don't just add to the cost; they multiply it with every step
6. **Debugging**
   In case of SQL query error - LLM will auto debug and re-run, taking up additional tool calls and costs.

Usage Patterns

The highest adoption I see is from operations, marketing, and finance teams. The following are the most common usage patterns from my client deployments

Operations, Marketing & Finance Teams

Natural language interface to backend datasets and uploaded CSV files

1. Pull specific customer and transactions records for review
2. Recon between finance and ops data
3. Insert / Update / Delete records
4. Download filtered data for offline analysis in Excel
5. Append fields and field cleanups
6. Generate summary reports with standard prompts for reuse
7. Generate PDF output

Many users prefer AI interface over their existing interfaces given the range of operations they can carry out and the efficiency of direct integration with automations

Analytics Folks

1. Pre-process raw tables and then download for offline analysis
2. Adhoc queries
3. Database level tasks requiring SQL

Platforms

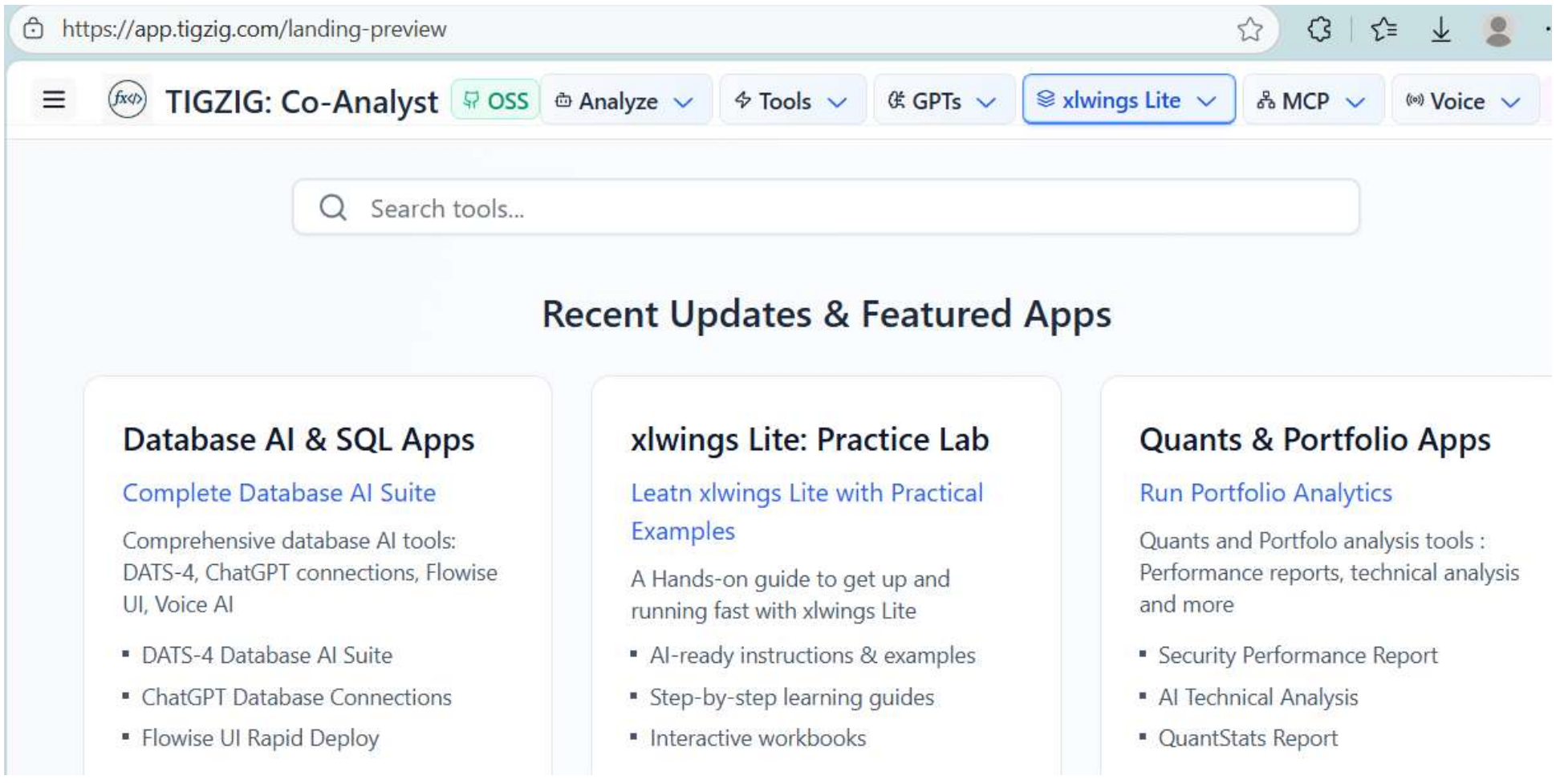
1. **Servers:** Very often client determined. But where you have a choice, here are my defaults
   Server based: Hetzner + Coolify for deployments. Allows a firewalled environment to deploy any apps and databases. Reliable performance and pricing.
   Serverless : Vercel for React & Render/ Railway for FastAPI
2. **Databases**
   - Neon: instant Postgres DB creation, deletion etc via API
   Top choice for AI apps requiring instant temp. databases
   - Aiven: great free tier.
   - Supabase: integrations esp. auth.
   - Standard / Self-Hosted: AWS RDS or Hetzner with Coolify
3. **SQLite** solid option for in-browser work. Requires setting up SQL Agent from the grounds up.
4. **LLM Gateway**: OpenRouter provides a single point gateway to all major LLM including the latest e.g GLM 4.5. Also, great set of reports to monitor costs.
5. **Custom User Interfaces**: React / NextJS / HTML-JS
6. **Backend Services**: FastAPI

2. Live Apps

DATS-4 : Deployed Live

app.tigzig.com

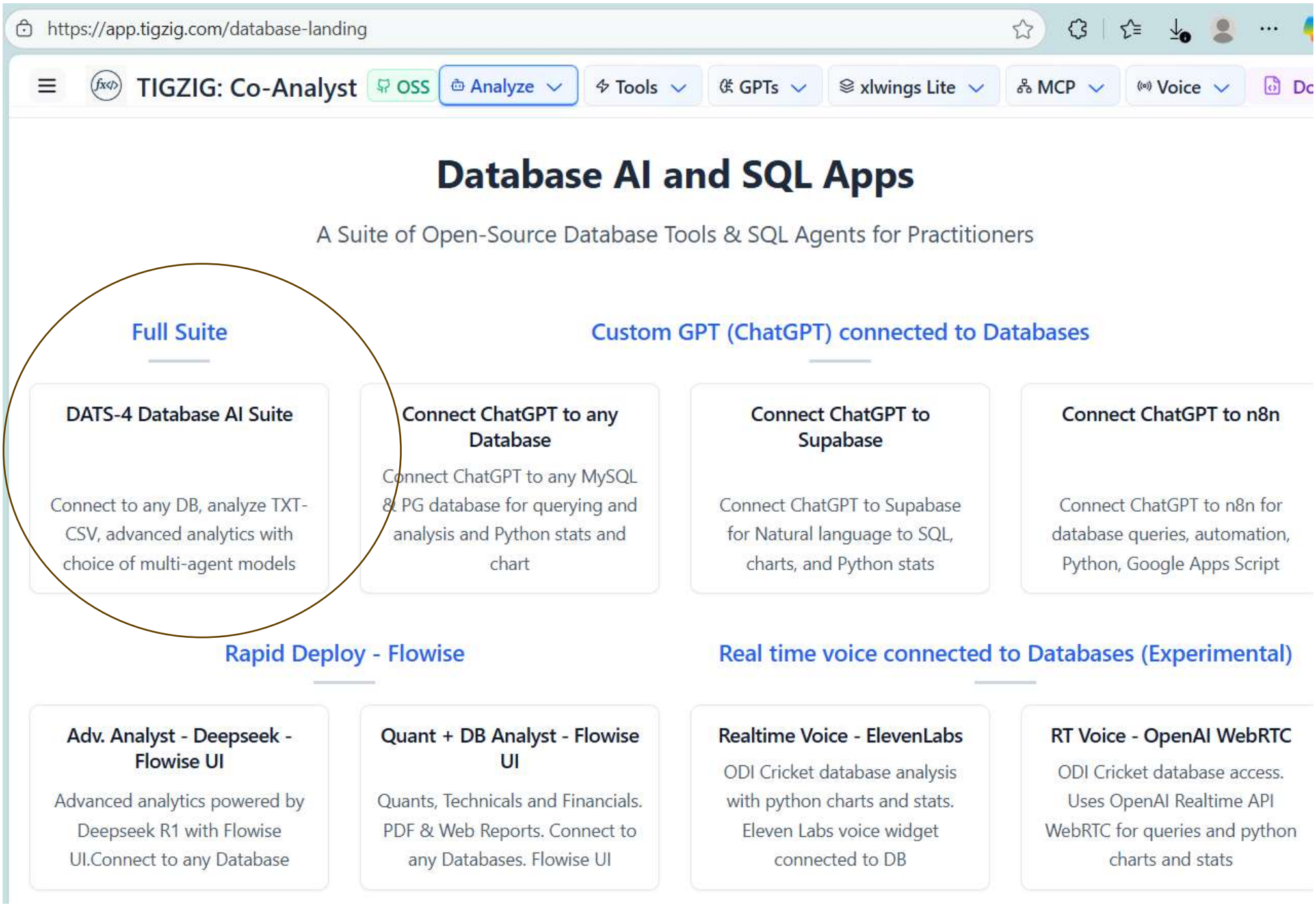
Path: Database AI & SQL Apps



4 Variants – 8 Live Apps

All apps live, fully functional and open source.

DATS-4 is the flagship app.



DATS-4: How to Use

Option 1 : Customize & Deploy

Option 2: Try Live on app.tigzig.com

1. Customize & Deploy

Open Source

Customize & deploy on your server/ VPNs

Core Deployment Patterns:

1. **Full Suite**: DATS-4 with custom UI.
2. **Custom GPT:** connected to the database backend.
3. **Rapid UI**: Flowise Agent UI for quick deployment.

Key Customization Areas

- **Security**
  - User API Keys / Oauth
  - Parameterized queries
  - DB user ID with restricted privileges
  - Row Level Security w/Postgres
- **Context** : Schemas, rows, queries, business rules
- **Interface** : customize based on user needs
- **Components**: deploy full suite or components
- **Functionality** : integrate additional functionalities

2. Try live on app.tigzig.com

The public site is a minimal-security sandbox configured to provide an unrestricted environment for testing the suite's full range of capabilities.

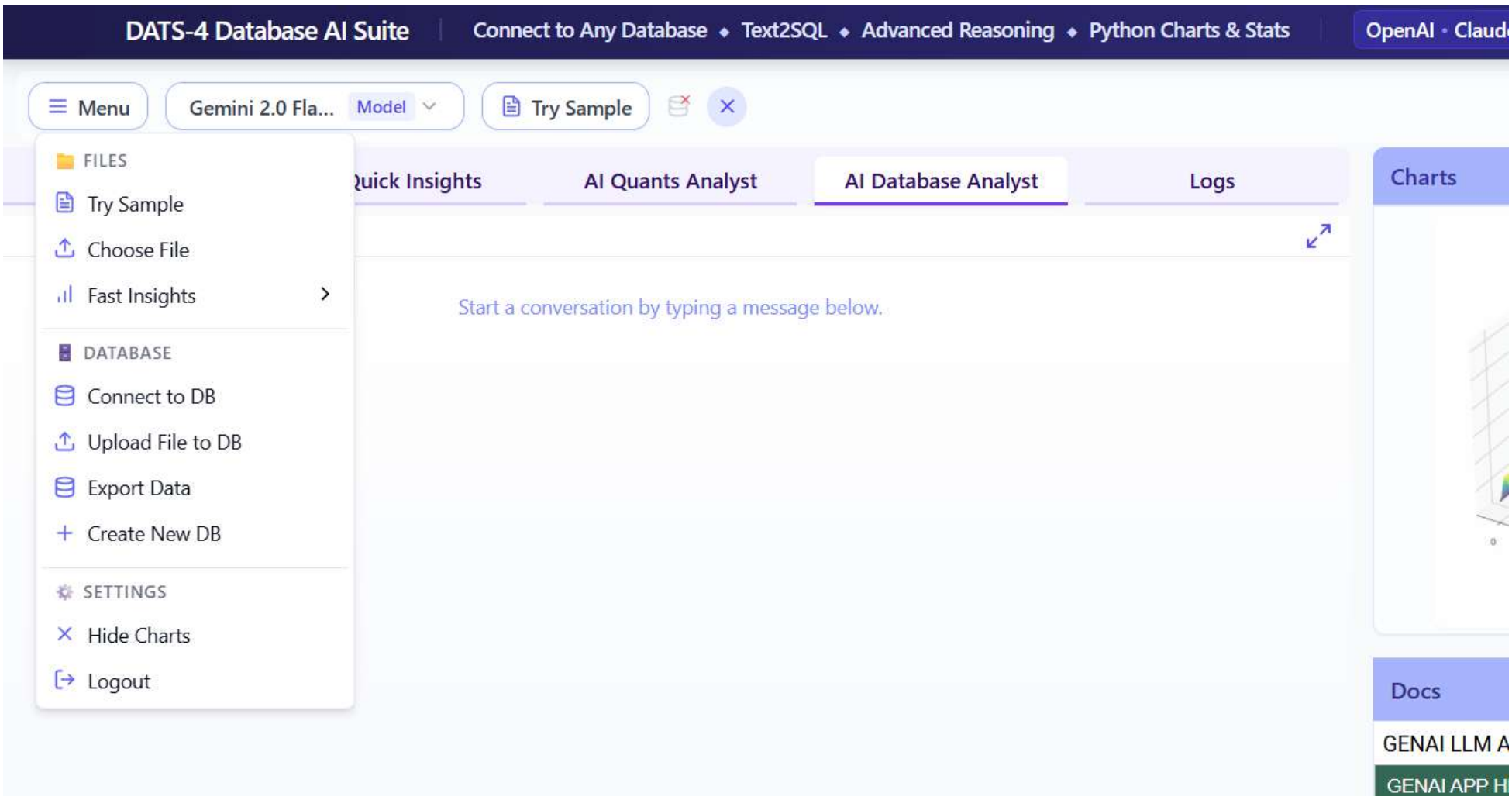
- Use on-the-fly temporary Postgres database generated by the app or create one instantly at Supabase/ Neon/ Aiven
- Use the sample files on the app / google drive

Practitioner's Warning

WARNING: All database credentials and queries submitted via the public app are logged on the backend. Use this sandbox with non-sensitive data and credentials only.

ADMIN-LEVEL ACCESS: The full DATS-4 suite is an admin-level tool with extensive logging. For end-user deployment, you must restrict functionality and customize logging configurations.

Interface Components



- 2 Agents : Main Database Analyst and Quants Analyst
- Sample data for rapid testing
- Menu option to upload files and connect to databases
- On-the-fly temporary Postgres database
- Choice of LLM for advanced analysis
- Chart & Document pane
- Logs
- File uploads: interactive grid and automated data quality metrics

Choice of LLM for Advanced Analysis

Choose your LLM for the reasoning step. The app setup also allows an efficient method to add and remove LLMs

Choose Advanced Analyst Agent Framework

Reasoning Model	Type	Quality	Cost	Select
Gemini 2.0 Flash	Best Value	75	Lowest	✓
GPT-5	Top Quality	95	Volatile	
Deepseek-R1-0158	Great Quality	90	Med	
Claude 4 Sonnet	Topmost Quality	100	High	
Gemini 2.5 Flash	Good	75	Low	
GPT-4.1	Great Quality	90	Med	
GLM 4.5	High Variances	80	Low	
Qwen3 235B Thinking	High Variances	75	Low	
Gemini Pro 2.5	Avoid	85	Highest	

Cost varies based on use case and AI setup. Refer attached price study for more details.

Open Price Guide

Cancel

✓ Confirm Selection

Core Workflows

1. **Sample File Test**

   Use the built-in sample data and an on-the-fly temporary database for rapid evaluation

2. **File Upload**

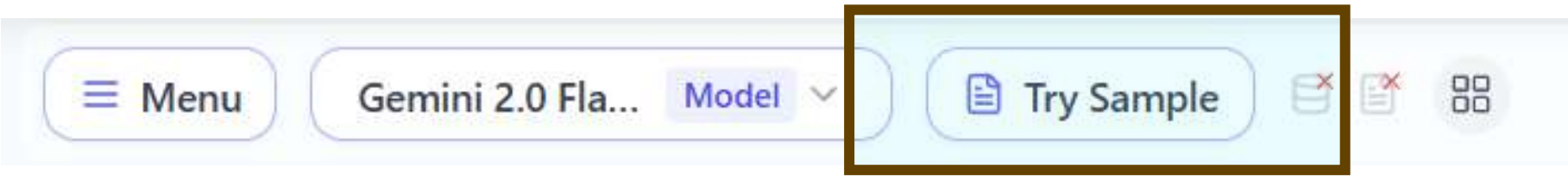
   Upload a local file (CSV/Tab Delimited) to a temporary or user-provided database.

3. **Direct Connection**

   Connect directly to a remote Postgres or MySQL database

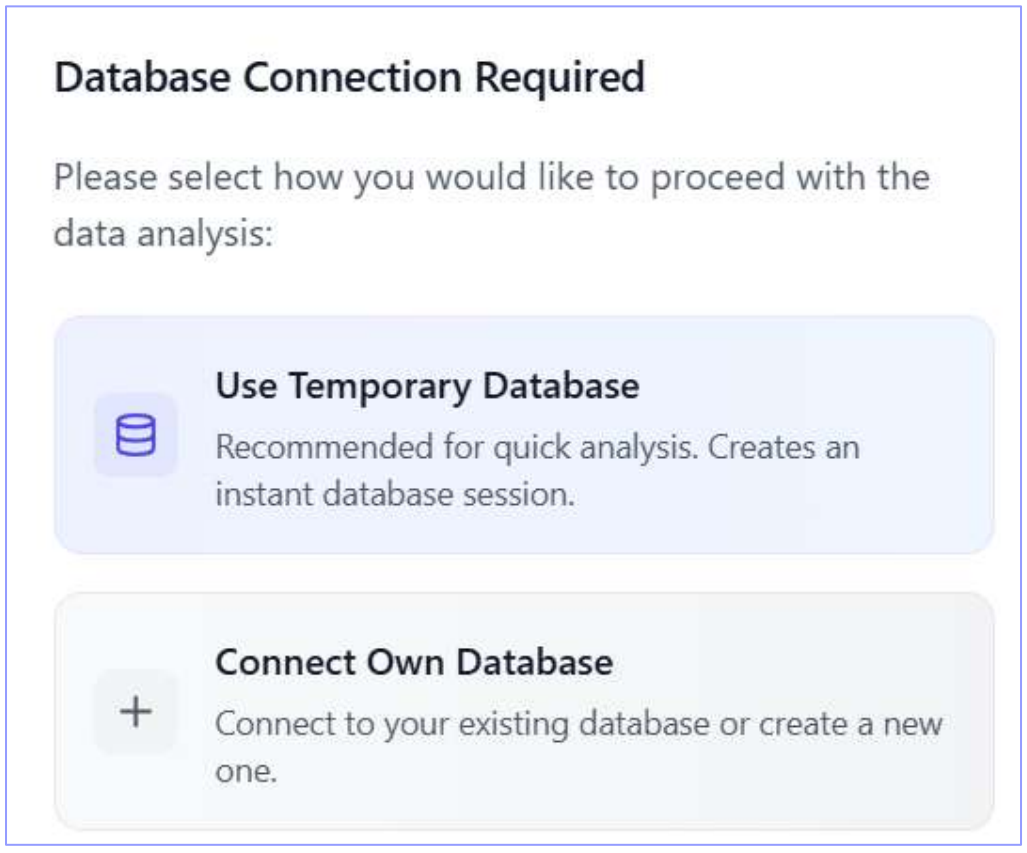
1. Sample File – Rapid Test

1.



2.

3.

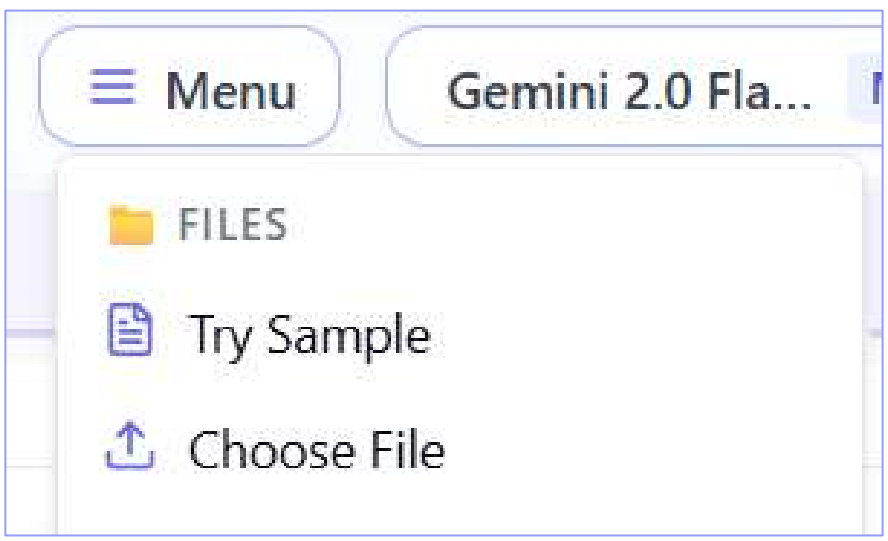


Use on-the-fly temporary Postgres database OR connect to your own DB.

4. Copy and paste ready to use starter prompt for quick analysis once database is setup. Or go with your own request.

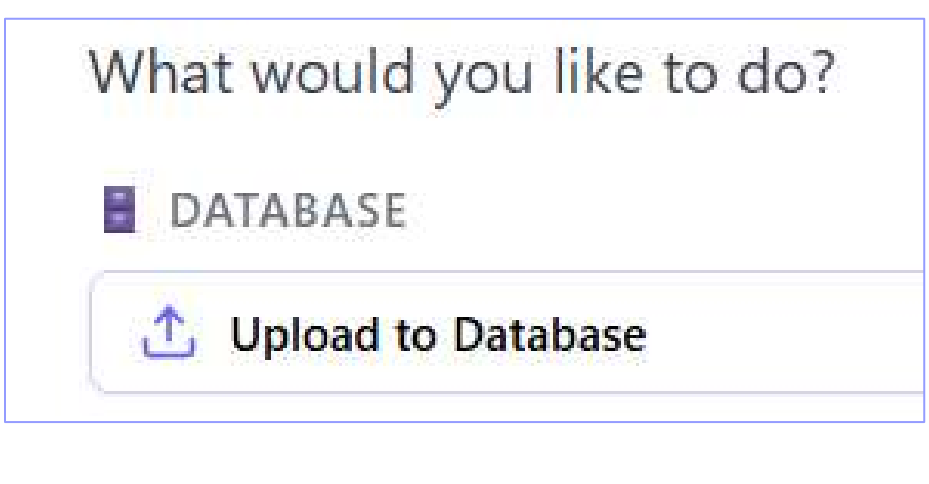
2. Upload Your File

1.

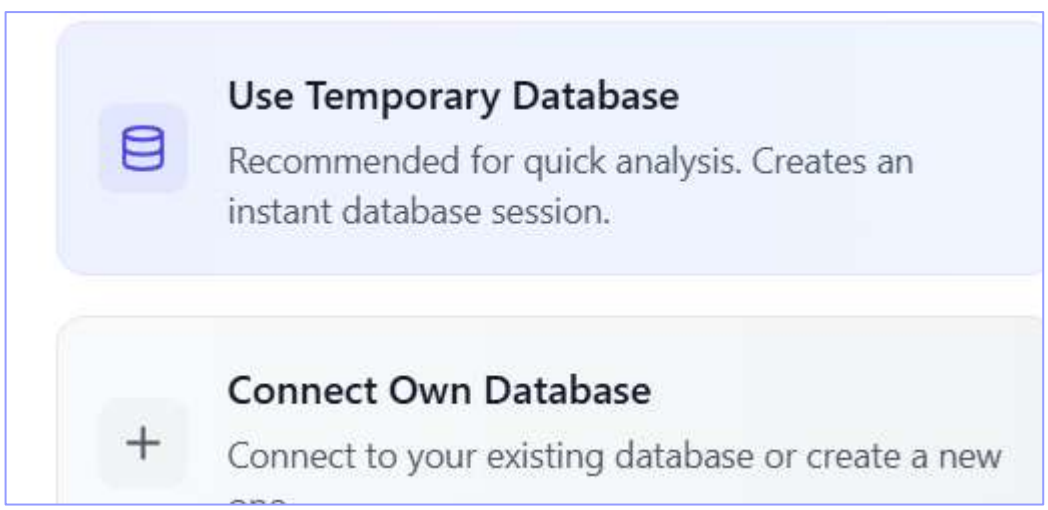


Select your file for upload. Supports CSV and tab delimited

2.



3.



Use an instant temporary Postgres DB OR connect to your DB

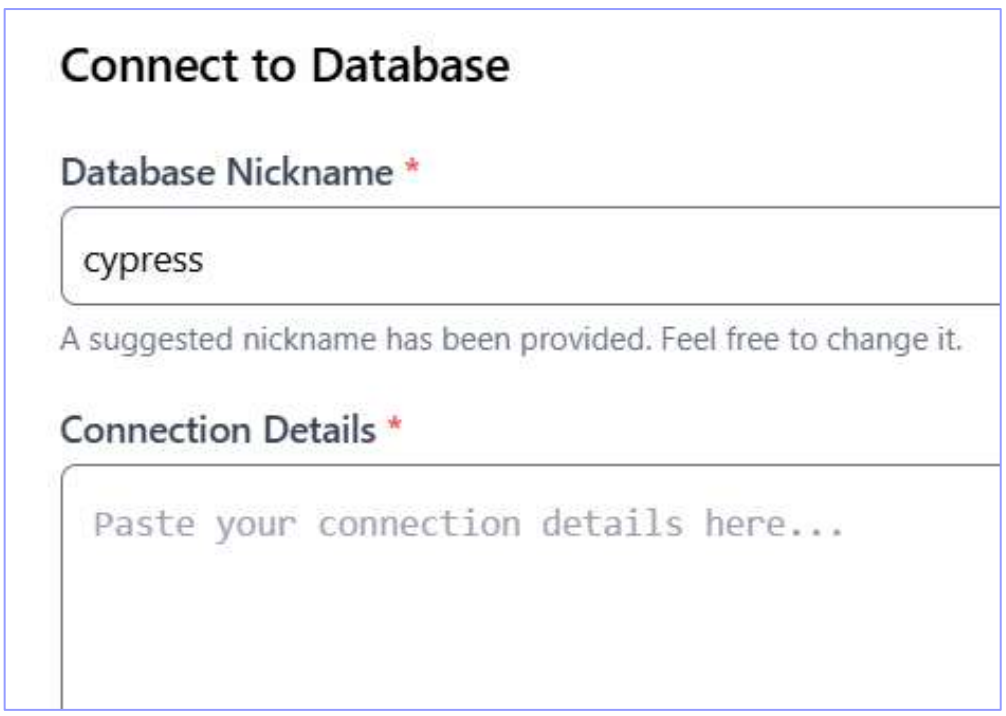
4. **File schema sent to AI automatically**. Go to Advanced Analyst tab and ask questions, analyze, create and customize charts – in natural language

3. Connect your Database

Postgres and MySQL supported

1. 

Menu -> Connect to DB

2. 

Paste your DB credentials. Format does not matter – URI / table / text – AI will parse it

3.
- Go to Advanced Analyst tab
- The AI agent does not automatically know your database schema upon connection. You must instruct it to list tables or query sample rows to provide it with the necessary context for analysis.
- Query, Analyze, Merge, Summarize, Visualize

Agent Reasoning View

Full trace of agent's reasoning process from business context and feature engineering to the final SQL queries and degug logs

Advanced Analyst

>_ Flow Step: llmAgentflow_0

Reasoning Process

Business Context & Analytical Goal:

The objective is to build a customer profiling data mart for advanced analytics and mach

1. **One record per customer** by merging demographic data (`BANK_UNQ_1M`) with t (`TRANS_10M`).
2. **Focus on transaction behavior**: Differentiate between cash (TC_CODE=1001) a transactions.
3. **Derived features**: Create up to 20 insightful va averages, ratios, recency, frequency).

Key Considerations:

- **Data Volume**: `TRANS_10M` has 10M+ rows. Ag
- **Feature Engineering**: Prioritize variables that
  - **Monetary behavior**: Total/average spendi
  - **Behavioral ratios**: Cash vs. retail preferenc
  - **Temporal patterns**: Recency (last transacti
  - **Variability**: Standard deviation of transaction amounts.
- **Technical Constraints**:
  - Use permanent tables (no temporary tables).
  - Limit to 6 SQL queries.

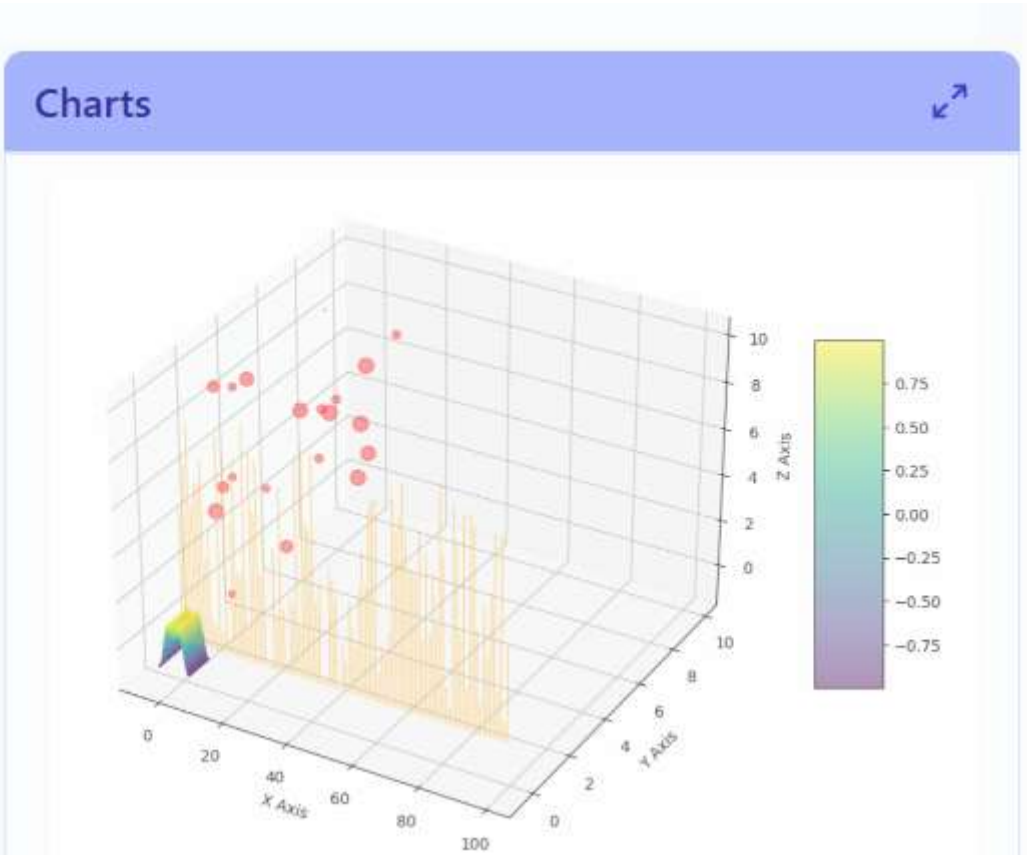
SQL Queries (MySQL)

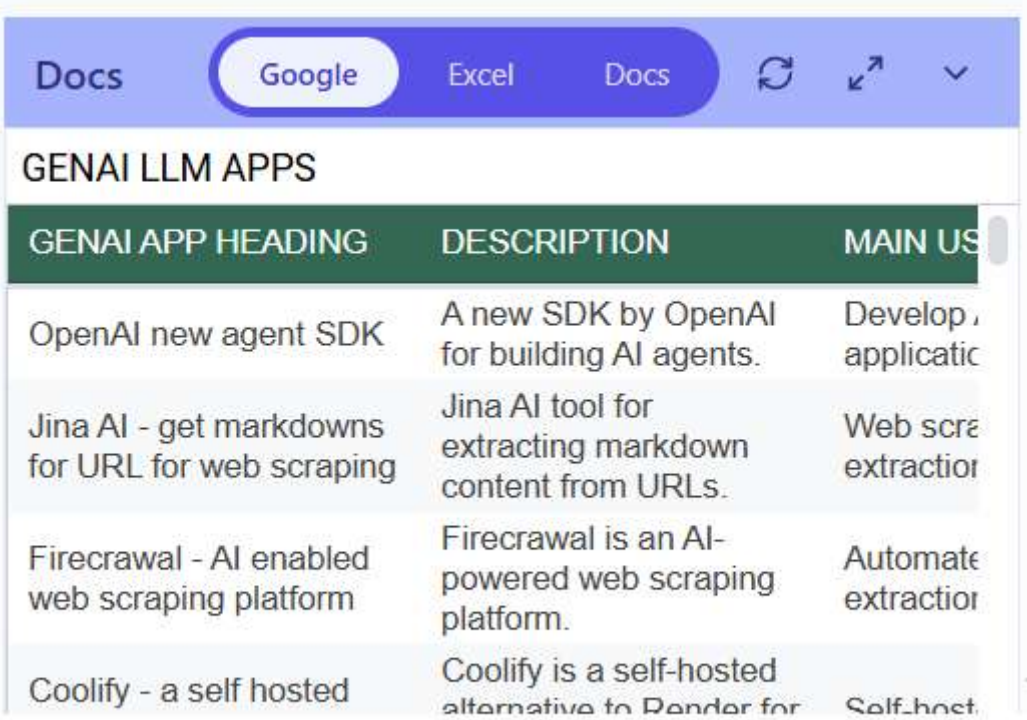
Analysis Queries (Data Mart Creation)

Query 1: Create Intermediate Transaction Aggregation Tab

```
CREATE TABLE IF NOT EXISTS trans_agg_temp AS
SELECT
    CUSTID,
    SUM(CASE WHEN TC_CODE = 1001 THEN AMOUNT EL
    SUM(CASE WHEN TC_CODE = 1001 THEN 1 ELSE 0
    AVG(CASE WHEN TC_CODE = 1001 THEN AMOUNT EL
```

Dedicated Charts & Docs Panel



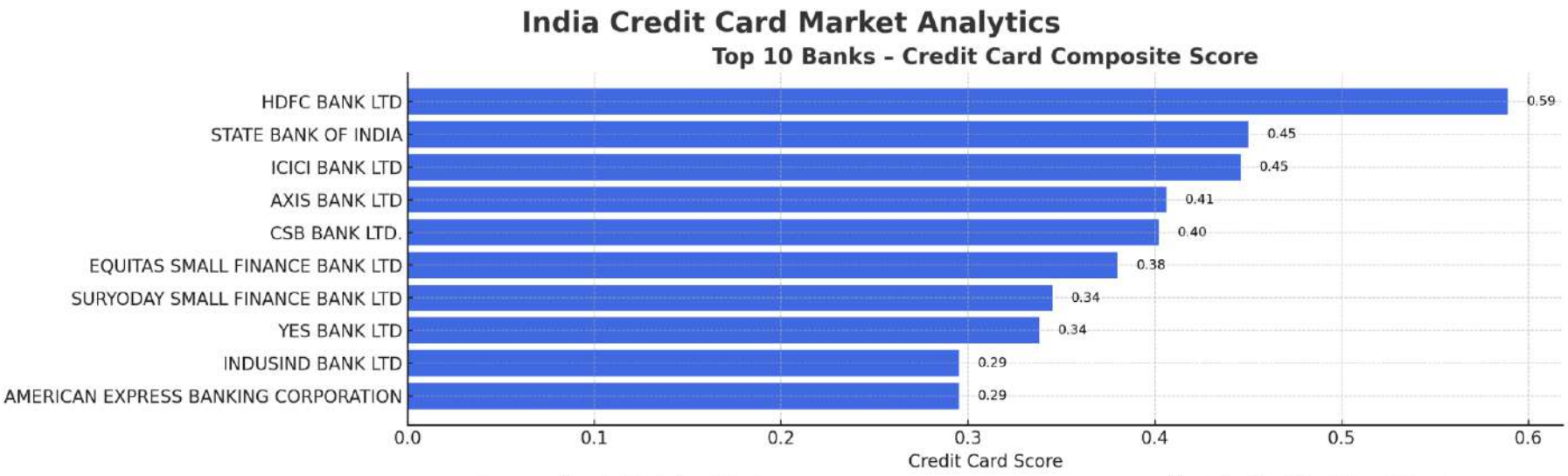


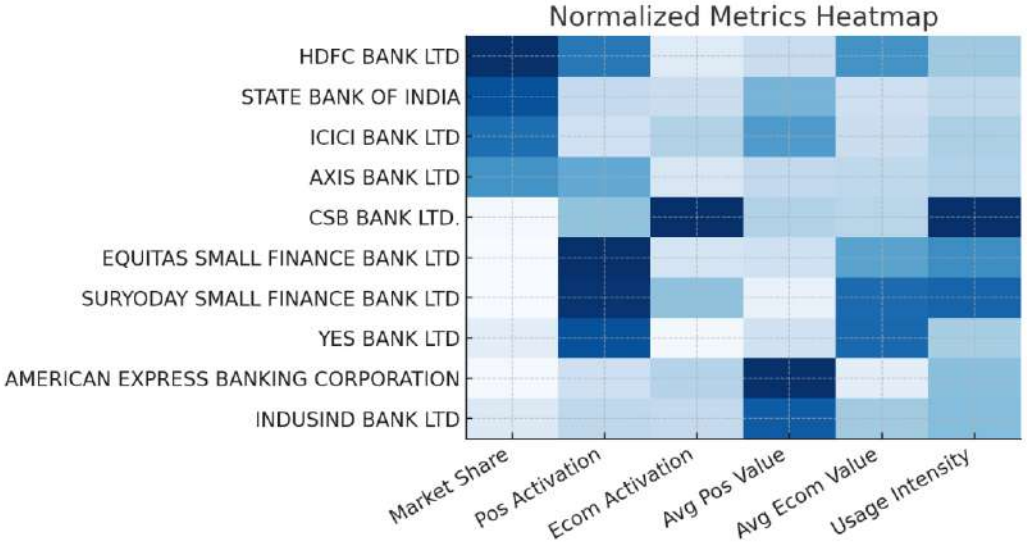
About • Privacy • Terms

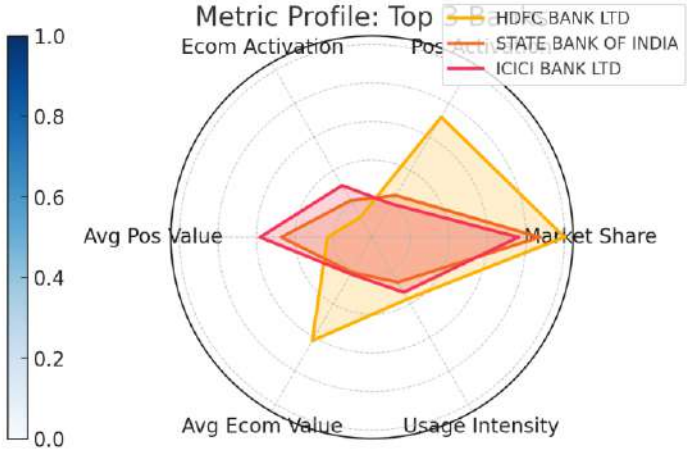
- Single-click toggle to open/close charts & document panel
- Dedicated, full-screen chart panel for visualizations.
- Dynamic document panel for live report and data updates.

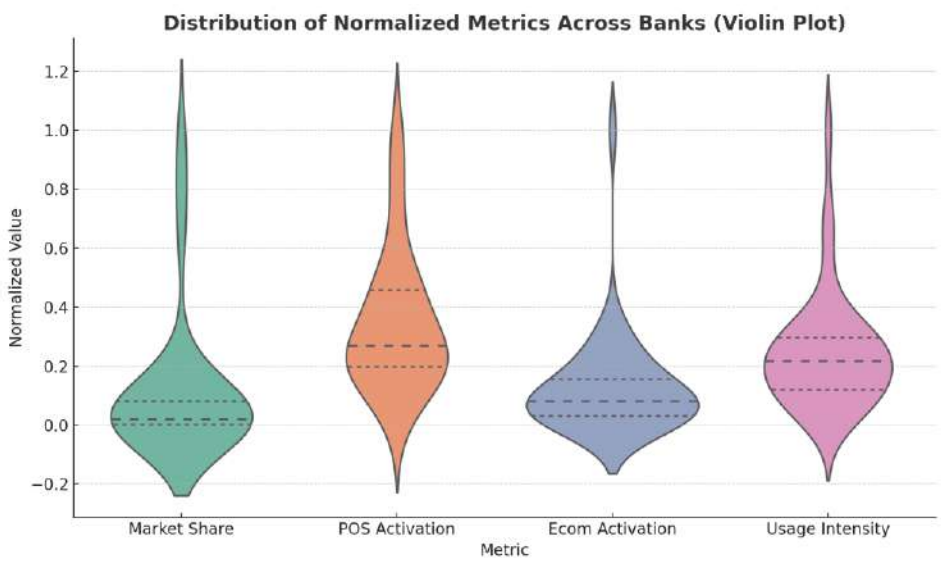
Python Charts

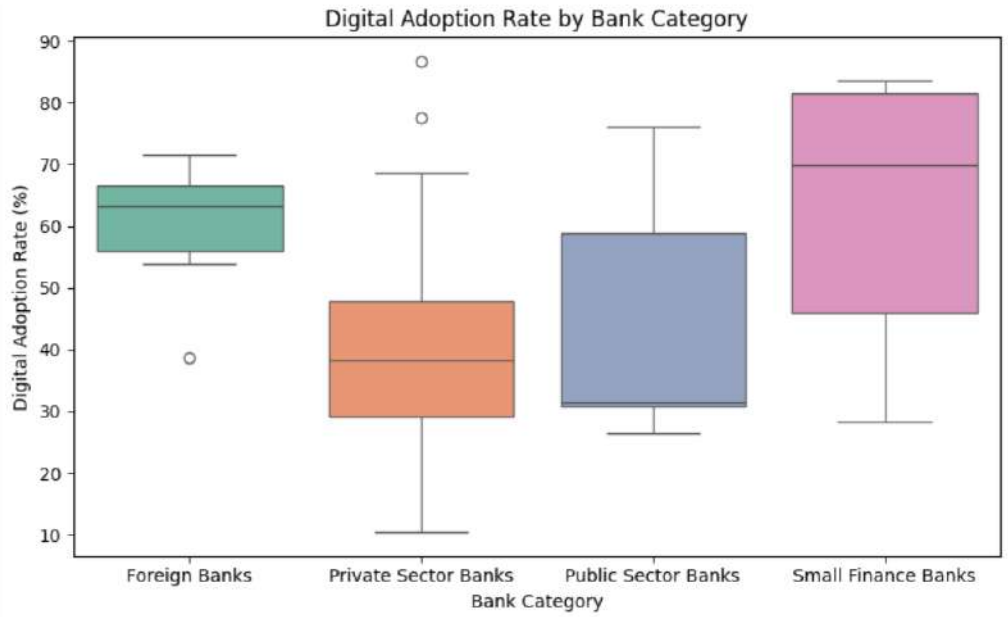
Integrated Python Interpreter for charting. All charts below were generated directly in the app.











Statistical Analysis

The integrated Python Interpreter enables full statistical analysis, not just charting.

View Agent Reasoning

Chi-Square Test of Independence: Education vs Job

Statistical Output Table

STATISTIC	VALUE
Chi-square	531,242.61
Degrees of Freedom	20
P-value	< 0.0001
Total Customers	918,208

Interpretation (Compact)

There is a statistically significant association between education and job (χ^2 = 531,242.61, df = 20, p < 0.0001
This means the distribution of job types is not independent of education level among customers.

View Agent Reasoning

ANOVA Test: Average Transaction Value by Education Level

ANOVA Summary Table

STATISTIC	VALUE
F-Statistic	34.21
P-Value	1.44e-15
Degrees of Freedom (Between)	2
Degrees of Freedom (Within)	285,714

Group Comparison Table

EDUCATION	N	MEAN	STD DEV	9 L
primary	56,467	9,700.87	1,687.74	9
secondary	170,401	9,634.69	1,812.47	9
tertiary	58,849	9,624.37	1,821.36	9

Interpretation

There is a statistically significant difference in average transaction value across education groups (p < 0.05).

PDF Output

- On-demand, formatted PDF report generation for all analysis and query outputs (text only)
- Report structure and content are fully customizable via natural language instructions.

RBI Credit Card Segment: Top Bank Rankings & Insights (March 2025)

Analysis Objective

To identify and rank the top 10 banks in India's credit card segment for March 2025, using a composite scoring system based on card issuance, transaction volumes, values, and usage intensity. The goal is to provide actionable insights into which banks are leading in both scale and customer engagement in the credit card market.

Methodology

1. Data Preparation & Derived Metrics

- Source: RBI monthly statistics for March 2025, table: rbicc_mar2025_78680.
- Key Metrics Derived per Bank:
- credit_cards_nos: Number of credit cards issued.
- total_txn_volume: Total credit card transaction count (POS + E-com + Others).
- total_txn_value: Total credit card transaction value (POS + E-com + Others).
- avg_txn_value: Average value per transaction = total_txn_value / total_txn_volume.
- usage_ratio: Transaction volume per card = total_txn_volume / credit_cards_nos.

```
rbicc_mar2025_78680
```

2. Normalization

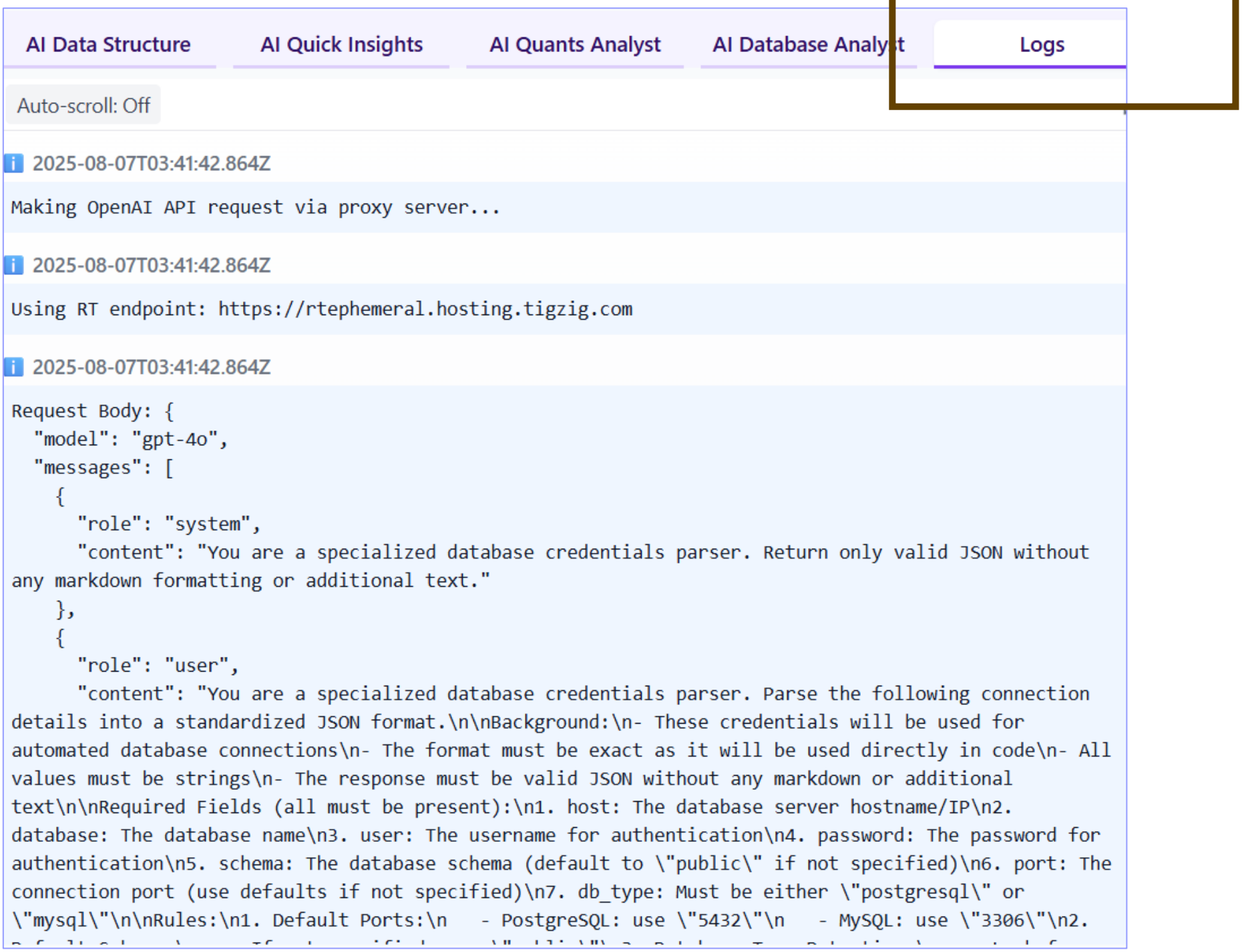
- Each metric is normalized using min-max scaling to [0,1] across all banks:
- norm_metric = (value - min) / (max - min)

3. Weighted Scoring System

- Weights assigned:
- Credit cards issued: 25%
- Transaction volume: 35%

Detailed Logs

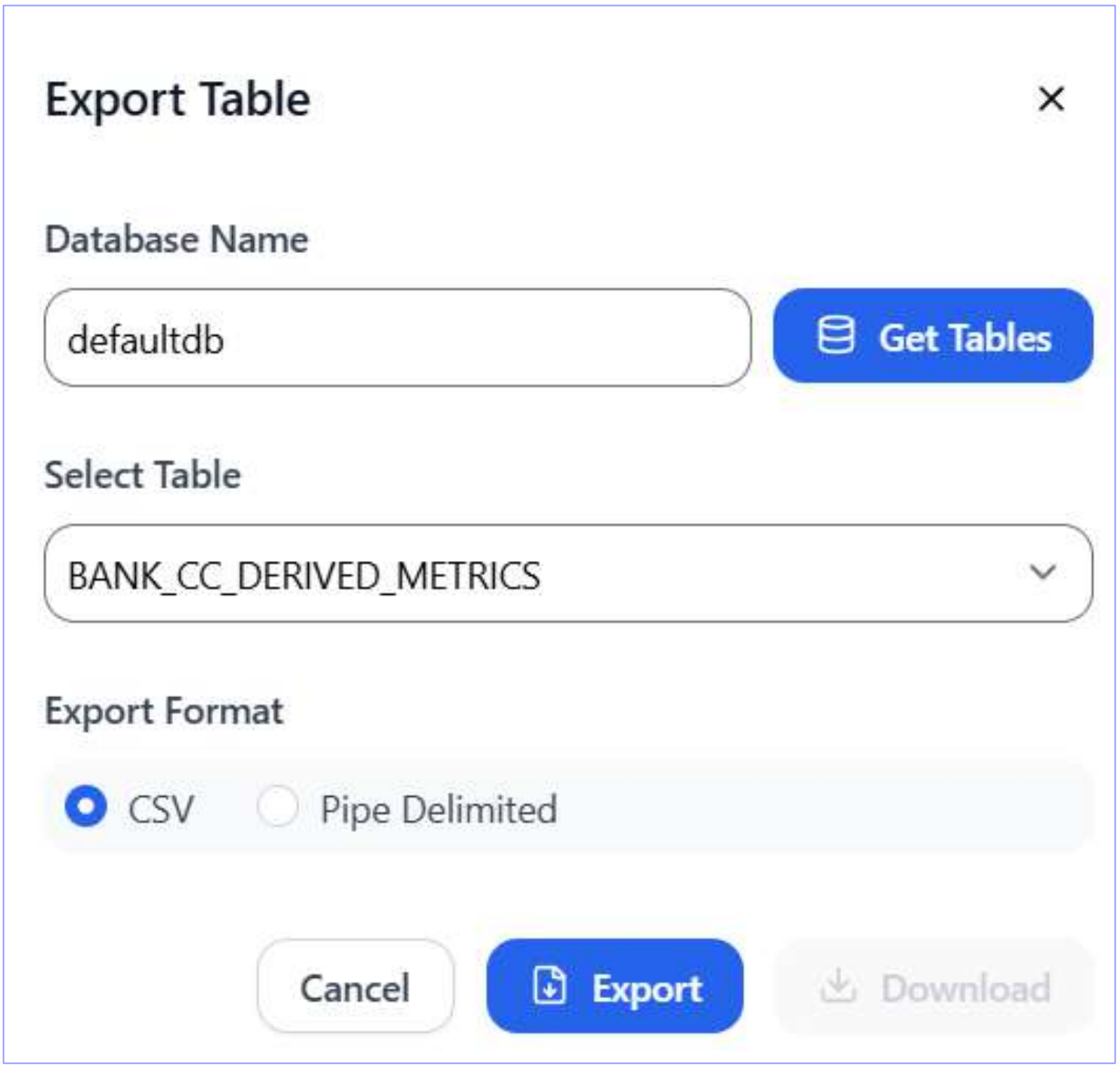
- Detailed logging of key API calls and actions
- Valuable for first line of debugging



Full logs are ‘admin’ level with sensitive info being logged. Restrict as per security access. For end users, remove / customize logging as per requirement

Export Tables

- Perform transformations and create new tables, then export any table to a local file (CSV or Pipe Delimited).
- Full support for both MySQL and Postgres environments.



Interactive Data Table & DQ Report

Data Table

Expand

Page Size: 10 | 1 to 10 of 64 | Page 1 of 7

BANK_NAME	ATM_CRM_ONSITE_NOS	CREDIT_CARDS_NOS
BANK OF BARODA	8,597	30,43,116
BANK OF INDIA	5,325	72,760
BANK OF MAHARASHTRA	2,150	27,869

- Interactive data grid for all uploaded files.
- On-the-fly descriptive statistics and data quality assessment.
- Record-level view with filtering and sorting capabilities.

Statistics for ATM_CRM_ONSITE_NOS

Calculations under validation & enhancement

Central Tendency

Mean	2,051.92
Median	431.5
Mode	0
Trimmed Mean	1,092.48

Distribution Plot

Percentile Distribution

Min P25 Median P75 Max

Dispersion

Variance	1,81,25,691.31
Std Dev	4,257.43
CV (%)	207.48

Distribution Shape

Skewness	3.783
Kurtosis	20.664

Position

Q1	0
Q2	
Q3	
IQR	
Range	

Percentiles

P10	
P25	0
P50	430
P75	1,936
P90	7,460

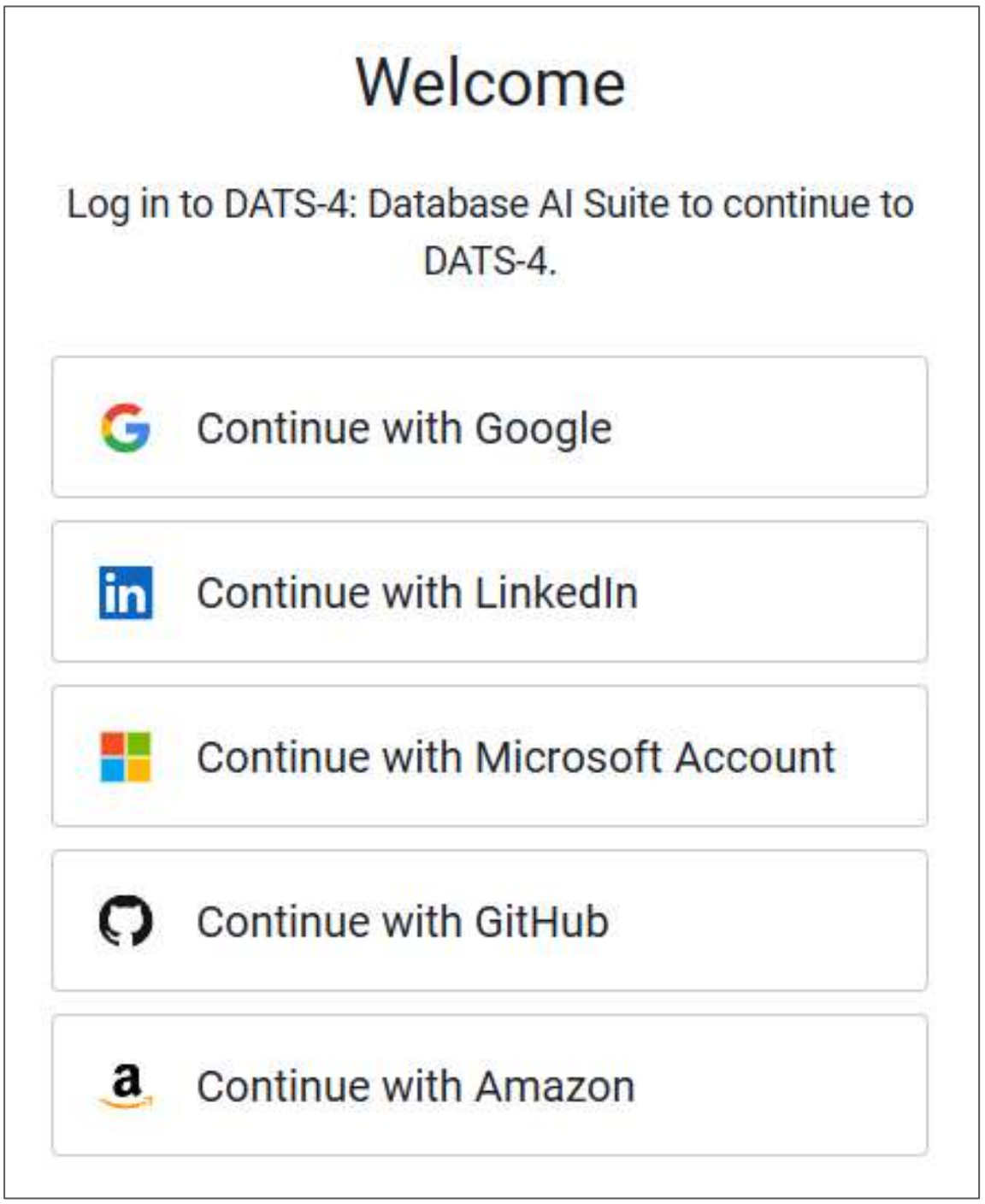
Data Quality

Unique Values	48
Missing Values	0
Outliers	9

BANK OF BARODA

CATEGORY	Public Sector Banks
DATE	2025-03-31
BANK_NAME	BANK OF BARODA
ATM_CRM_ONSITE_NOS	8,597
ATM_CRM_OFFSITE_NOS	2,390
POS_NOS	52,652
MICRO_ATM_NOS	45,244

SSO with OAuth



- OAuth-based single sign-on (SSO) via Auth0 for - Google, LinkedIn, Microsoft, GitHub and Amazon.
- Current Scope: The baseline implementation is linked to ‘Create DB’ function only. This provides unrestricted testing of analysis tools without forcing an app level login

This baseline setup is built for extension. It provides the OAuth foundation needed for full app authentication, fine-grained access controls, and row-level security in a live client project.

App Variants

There are three stable variants of the app, each suitable for a different use case . The fourth, Voice AI, is experimental.

1. **Custom GPT**
   - Custom GPT connected to databases is a robust, effective solution - straightforward setup and low maintenance
   - Combines a front-end UI, built-in AI Agent, and the full native ChatGPT feature set
   - No separate API Cost for Agent + GPT-5 access
   - Efficient to connect automation backends and other apps via FastAPI/ n8n / Flowise / Make.com
   - **This is my first choice**

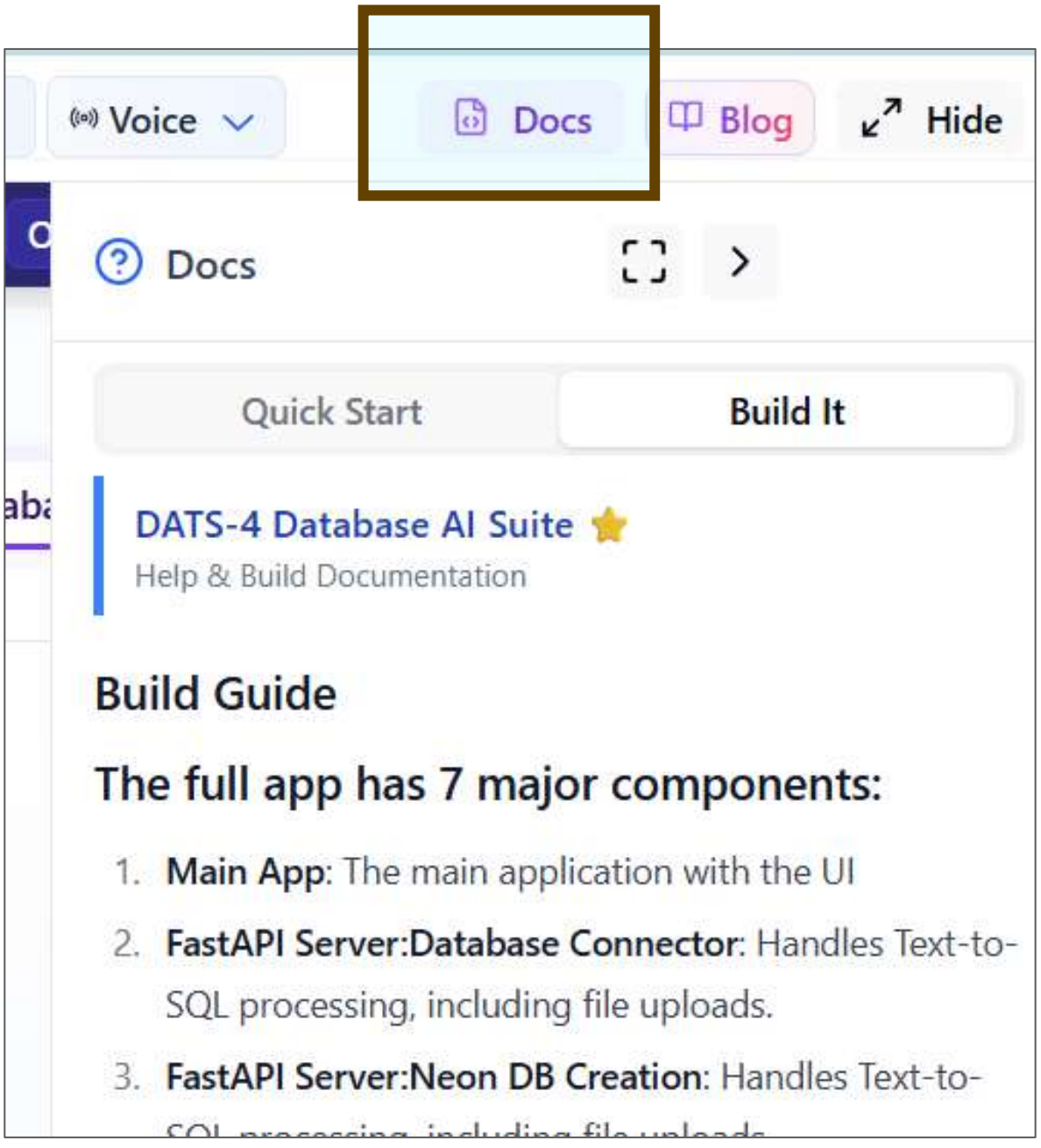
2. **Flowise / n8n**
   - Built-in user interfaces from Flowise and n8n.
   - Setup is efficient with direct connection to automation backends and other apps
   - API Charges - as per usage

3. **Database AI Suite -4**
   - Top choice where full feature and customization needed
   - Fully customisable : user interface as well as backend
   - Deploy anywhere, connect to Oauth
   - API Charges - as per usage

3. Source Code

Source Code

All source codes links on app page in docs section



- Hit Docs on the app page

- Links to source codes and build guide, including video guides

GitHub Repositories

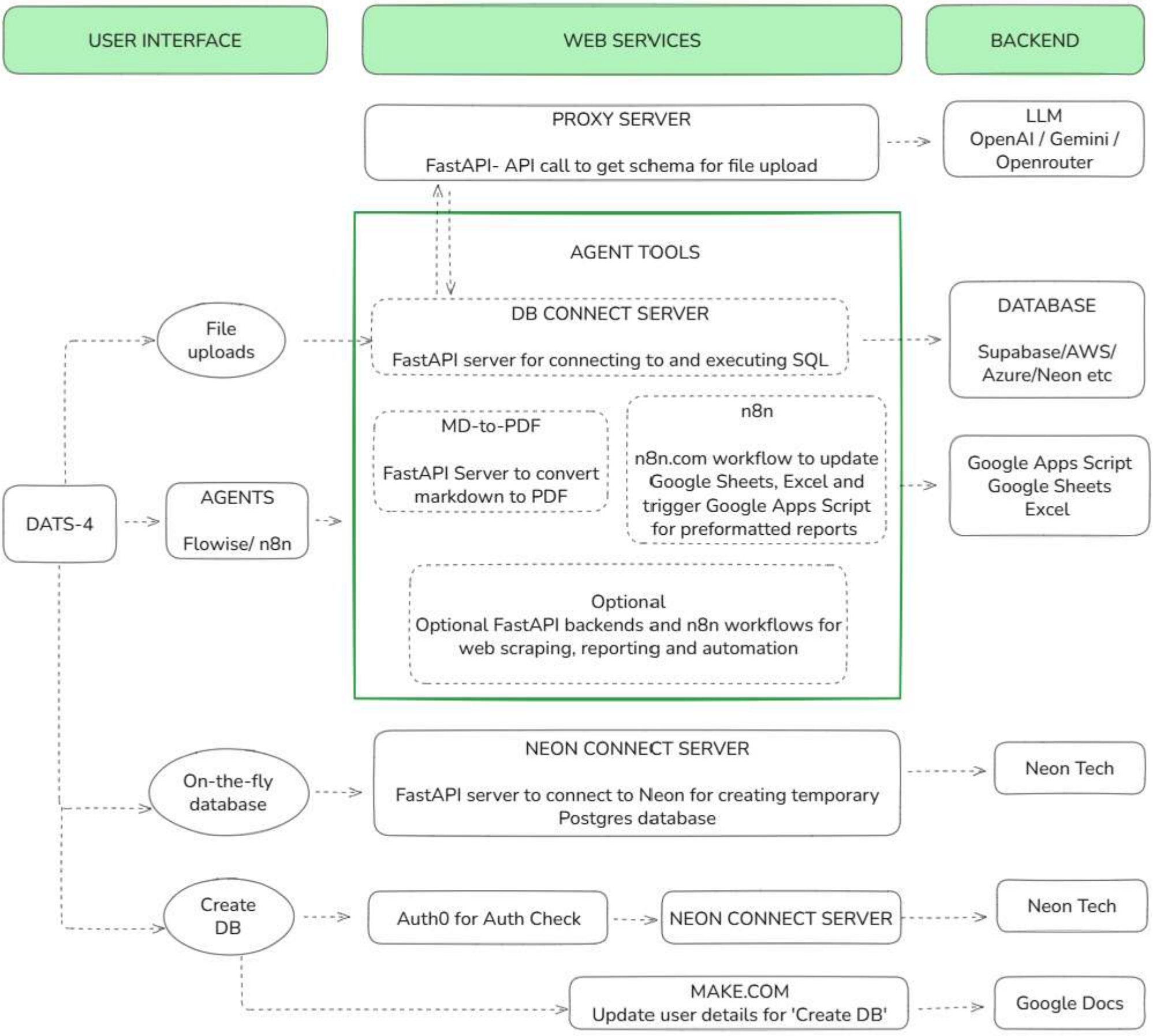
1. Main Repo
   https://github.com/amararun/shared-rexdb-auth-embed-v3-agentflowv2

2. FastAPI Server : SQL DB Connect
   https://github.com/amararun/shared-fastapi-rex-db-coolify

3. FastAPI Server: Neon Database

Architecture

Modular architecture for efficient integration of automation services or backend



Components Based Architecture

- Mix and match deployment of individual tools
- Connect components to your own user interface

GitHub Repos	Description
Main App	The main application with the UI
FastAPI: Database Connector	Handles Text-to-SQL processing, including file uploads
FastAPI: Neon DB Creation	Temporary database creation with Neon
Flowise Agent Schemas	Sequential Agent Framework with LLM Agent built with Flowise AI
Proxy Server	For API Calls to OpenAI / Gemini / Openrouter
MCP Server - Markdown to PDF	For converting markdown to PDF
Quant Agent Backend Repos	The TIGZIG Quants Agent app integrated into a single tab

Numerous more components available open source at app.tigzig.com - web scraper, pre-formatted slide deck creator, Excel table to PDF, Excel and Google Sheets updater, file converters, finance data extractors...

Architecture Overview Doc

This is my personal app architecture file for DATS-4 that I feed to AI Coder at start of every session. Enables immediate productivity without full codebase exploration. Includes critical gotchas from earlier experiences.

DATS-4 - AI Coder Documentation

Architecture Overview

Tech Stack: React 18 + TypeScript + Vite + TailwindCSS
Deployment: Vercel (can run in iframe or standalone)
Authentication: Auth0 (iframe limitations handled)
Data Processing: File upload → AI analysis → Database storage → Visualization

Key Dependencies

```
Core: react, typescript, vite, tailwindcss
UI: @radix-ui/* (dialogs, dropdowns, tooltips), lucide-react (icons)
Data: ag-grid-react (tables), axios (HTTP), uuid
Auth: @auth0/auth0-react
Styling: tailwindcss-animate, class-variance-authority, clsx
Markdown: react-markdown, react-syntax-highlighter
```

Project Structure

```
frontend/src/
├── App.tsx                    # 🔥 MAIN APP - All state &
orchestration (2094 lines)
├── components/
│   ├── ui/                   # Shadcn-style base components
│   ├── file-upload-section.tsx     # File selection handling
│   ├── analysis-tabs.tsx           # Main tabbed interface
│   ├── chat-box.tsx                # AI chat interface
```

README

All GitHub repos with README with step-by-step guide

DATS-4 Database AI Suite

Live App

A full version of this app is deployed and available at app.tigzig.com

Documentation

- **User Guide** - Comprehensive guide on how to use the application
- **App Architecture** - Detailed explanation of the application architecture
- **Setup Guide** - Continue reading below for installation and setup instructions

About The Project

The DATS-4 Database AI Suite is a tool for AI-driven data analysis against relational databases. Its primary function is to connect to remote PostgreSQL or MySQL databases to execute queries. Queries

Getting Started

Follow these instructions to set up and run the project locally.

1. Clone the Repository

First, clone the repository to your local machine using the following command:

```
git clone <https://github.com/amararun/shared-rexdb-file-upload> .
```

Note: Check for the latest shared repository name in case it has changed.

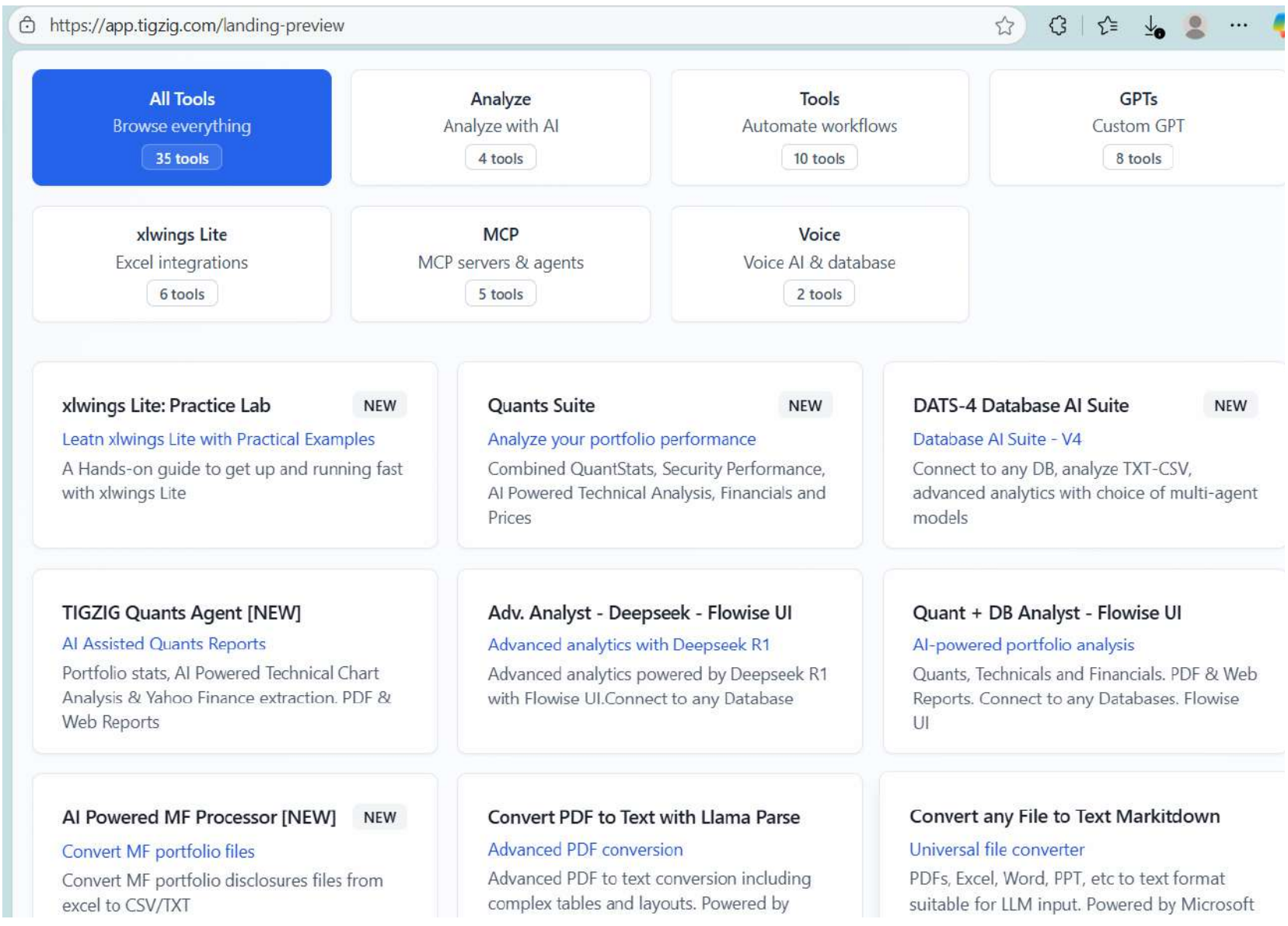
2. A Note on Analytics Tracking

TIGZIG: Micro-Apps for Analytics

25+ apps: Database AI / xlwings Lite / Automation / Quants

Access the full suite of open-source tools at

app.tigzig.com



Amar Harolikar

Specialist - Decision Sciences & Applied AI

Builder of app.tigzig.com