JAKUB BŘEZÍK

DEVELOPER · DATA AND API INTEGRATION

Prague, Czech Republic
+420 775 611 501
jakub@brezik.net
brezik.net
linkedin.com/in/jakub-brezik



SKILLS

Integration

REST API · OpenAPI · Informatica BDM · Informatica MDM/RDM · Informatica PowerCenter · Stonebranch UAC/UDM

Programming

Java · Spring Boot · Perl · Python · PL/SQL

Tools

BTEQ · Apache Velocity · Maven

Cloud

AWS · EC2 · S3 · ALB/WAF · Cognito · Lambda

Databases

Teradata · Oracle · PostgreSQL

DevOps and CI/CD

Jenkins Pipeline · Docker · AWS CloudFormation · Git · Unix

AI

Claude Code · AI agents · Spec-driven development

Collaboration

Jira · Confluence

Modelling

PowerDesigner

SUMMARY

Developer with 20 years in IT, the last 10 in data warehousing and data integration between systems in banking and insurance. From 2018 to 2026 at KBC I did the reference-data integration from source systems to data warehouse: design, build, orchestration, testing and daily operations across four environments and ten KBC Group entities. I design asynchronous, file-based integration flows with validation and fallback mechanisms, and build services around them. I make complex systems easy to see through, in data, tools and documentation.

EXPERIENCE

INTEGRATION & ETL DEVELOPER, REFERENCE DATA

KBC (via Adastra) | 2018 – 2026 · Prague

Project Rainbow: KBC Group's central data warehouse on Teradata (FSDM model) for internal and regulatory reporting, including reporting to the ECB. It serves ten local and group entities. End-to-end owner of the reference data flow from RDM to the Teradata DWH. Codelists are enablers: without fresh and correct values the rest of the warehouse cannot load reliably.

Architecture and design

- Implemented Informatica BDM export mappings on top of the existing canonical format, a view that flattens RDM business objects into single table, so one mapping serves every codelist and scales well.
- Proposed reversing the data flow between environments (production to integration, integration to development) for the internal RDM data-testing case that relied on live data, defended it before the technical committee, got it approved and negotiated the routing with the platform and security teams.
- Introduced OpenAPI specifications and swagger-ui for the data testing REST services; the RDM application team adopted the same setup for their own interface.
- Maintained the integration documentation and the content and domain ownership catalogues in Confluence, used by the team, analysts and DWH operations.

Integration engineering

- Owned the export chain: denormalizing view on the RDM database, Informatica BDM mapping, flat-file export, file transfer to the Unix servers, ingestion and loading through the database layers on Teradata.
- Replaced the standard one-job-per-table (400+ jobs) load with a looping job that validates each table, loads the ones that pass, stops on nulls or structure mismatches.
- Wrote the housekeeping modules: archiving and retention of exports, RDM housekeeping through its REST API, and the data-testing checks that call the in-house testing service.

LANGUAGES

Czech (native)

English (proficient)

CERTIFICATIONS

AWS Certified Cloud Practitioner
2021 – 2024

Informatica Developer Tools for Big Data Developers; Big Data for Developers
2018

Informatica PowerCenter 9.x Developer, Level I and II
2016

Teradata Advanced SQL
2015

EDUCATION

Masaryk University, Faculty of Informatics, Applied Informatics (not completed)
2005 – 2006

SOŠ a SOU Spojů, Tábor — electrical engineering, telecommunications and computer technologies
2001 – 2005

HOMELAB

Proxmox and Docker at home: trying new technologies, running a development and test environment for demos, and isolated game servers my kids can play on safely.

Proxmox | Docker

- Maintained source file XML definitions, normalization ETL and the loading transformations stored in PostgreSQL.

Services and tooling (Java, Spring Boot)

- Data-testing service: compares the exports with the RDM source and the Teradata target and exposes the checks as a REST API. It replaced a manual comparison in Excel that took tens of minutes per query run, down to seconds. Results are wired into Jira Zephyr test suites and Confluence tables through their APIs.
- Environment generator: creates integration, acceptance and production variants of an entity's roughly 1,000 template objects, including flows and orchestration, so one development environment behaves like three environments. A single-entity sandbox load took about 90 minutes and had to be wiped between datasets; migration tests now load all three at once and analysts compare live. Test suite runtime went from days to hours, allowing multiple loads per day.
- Operations dashboard: runs the check SQL on PostgreSQL and posts a compact status grid (three environments × ten entities × layers) to Confluence after each daily load, scheduled by Jenkins. Each tile expands to defect details and links. Covers database space, missing objects, column drift, nulls, load delays, soft errors, new and obsolete tables, volume spikes and empty loads, cutting detection and fix times. Handed over to DWH operations.
- Offline codelist viewer: converts the exports into one Excel workbook with an index and one tab per codelist, with ownership metadata, for users without RDM access. Introduced delivery of generated codelist workbooks to the organization's SharePoint through the managed file-transfer platform.

Operations and DevOps

- Daily operations of the reference-data loads on all environments. Codelists are enablers for the rest of the warehouse, so failures were diagnosed from Unix, Teradata and PostgreSQL logs and fixed immediately.
- Wrote my own Jenkins pipelines for builds, deployment and operations jobs.

Java | Spring Boot | REST | OpenAPI | Informatica BDM | Informatica MDM/RDM | Stonebranch UAC/UDM | Teradata | Oracle | PostgreSQL | Perl | Python | Jenkins | Jira | Confluence | Unix

DWH DEVELOPER

Komerční banka (via Anywhere) | 2017 – 2018

Internal DWH enhancements on Teradata: extended the HR table models and workflows, extended and refreshed the KPI transformations for the next year.

Teradata | BTEQ | PowerDesigner

BPM DEVELOPER

IBM (contractor) | 2017

Refactored the core business module and implemented processes on the IBM BPM platform.

MS SQL | IBM BPM | JavaScript

DWH DEVELOPER, IBFS

Société Générale (via Accenture) | 2016 – 2017

Built Teradata warehouses for the non-French countries from templates: parametrization and deployment per template, modelling of the country-specific extras in PowerDesigner. Introduced Apache Velocity for template-driven code generation.

Teradata | BTEQ | PowerDesigner | Apache Velocity | PowerShell

ETL DEVELOPER, SOLVENCY II

NN (via Accenture) | 2015 – 2016

ETL transformations in Informatica PowerCenter on Oracle and data modelling in PowerDesigner. Functional and technical design, defect analysis, code reviews, unit tests, test scenarios and support of system and UAT testing.

PowerDesigner | Informatica PowerCenter

EARLIER EXPERIENCE

CSC, SkillZone | 2006 – 2014

Customer support, project coordination of application deployments and server migrations, knowledge management and reporting, network and systems administration.

Remedy | ServiceNow | MS SQL | IBM Cognos

PROJECTS AND AI

AI is part of my daily toolkit: analysing errors and logs, writing scripts and tests, reviewing code. Every output is checked by tests and review, so the speed does not cost accuracy.

ARCANISTA – COLLECTION MANAGER FOR MAGIC: THE GATHERING

Personal project (in progress)

Web app for TCG collectors: what you own, where it is stored, what it is worth and what is missing from a set or deck. Product, UI/UX, architecture and build on my own, with AI agents as the development team.

AI-driven development

- I run Claude Code agents as a team: an approved spec, tests written from it in one session, the build in another that may not change them. CI enforces it.
- Parallel agents in separate git worktrees pick up GitHub issues and open pull requests; I decide and approve.
- Rewrote the app from Django/Python to TypeScript in two months: 1,100+ commits, 170 pull requests, 220 test files, 37 architecture decision records.

What sets it apart

- Every physical card is its own copy with condition, language, price paid and location, not just a quantity.
- Real storage: boxes, binders and folders nested to any depth, so you know where each card is.
- Phone scanning: cards recognised from a photo by image embeddings, learning from corrections; nothing is written until you confirm.
- Prices from Czech, European and US shops with history; sources are never blended, a missing price is never shown as zero.
- Every edit can be undone, deletes go to a recycle bin, and it handles like a desktop application.

Cloud

- Development environment on AWS as infrastructure as code (CloudFormation): EC2, ECR, ALB with WAF, Cognito, Route 53, S3, SES with Lambda, a budget alarm. Deployment through OIDC, with no stored keys.
- Deploying takes a couple of clicks; between test rounds the app runs locally on Docker Compose to keep the cloud bill down.

Claude Code | TypeScript | Node.js | Hono | React | TanStack | Vite | PostgreSQL | pgvector | Drizzle | ONNX Runtime | Docker Compose | Vitest | Playwright | Storybook | AWS CloudFormation