ForkJoin

Performance Optimization

Fintech | Trade Operations | Financial Trading Platform

We Advise. Develop. And Deliver.

ForkJoin

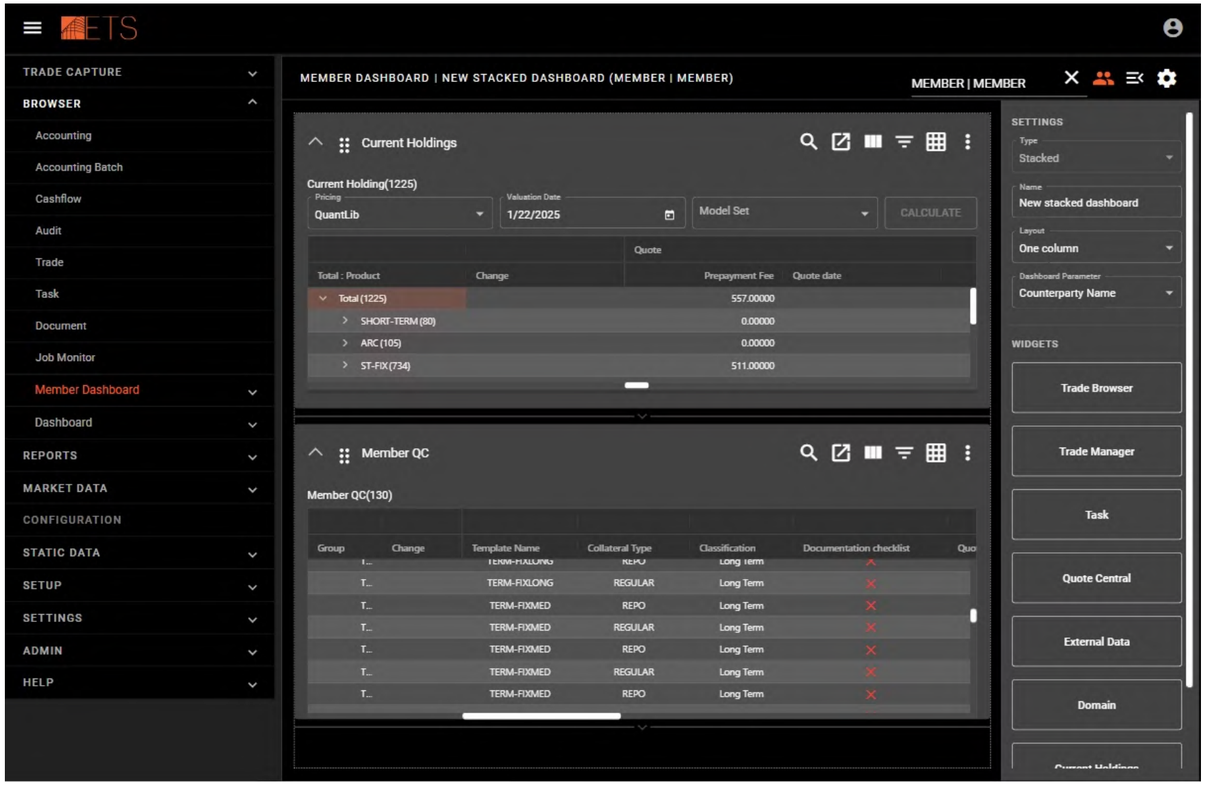
This project involved a **comprehensive financial trading platform** designed to support the full lifecycle of trade operations - from trade entry and workflow automation to reporting and compliance.

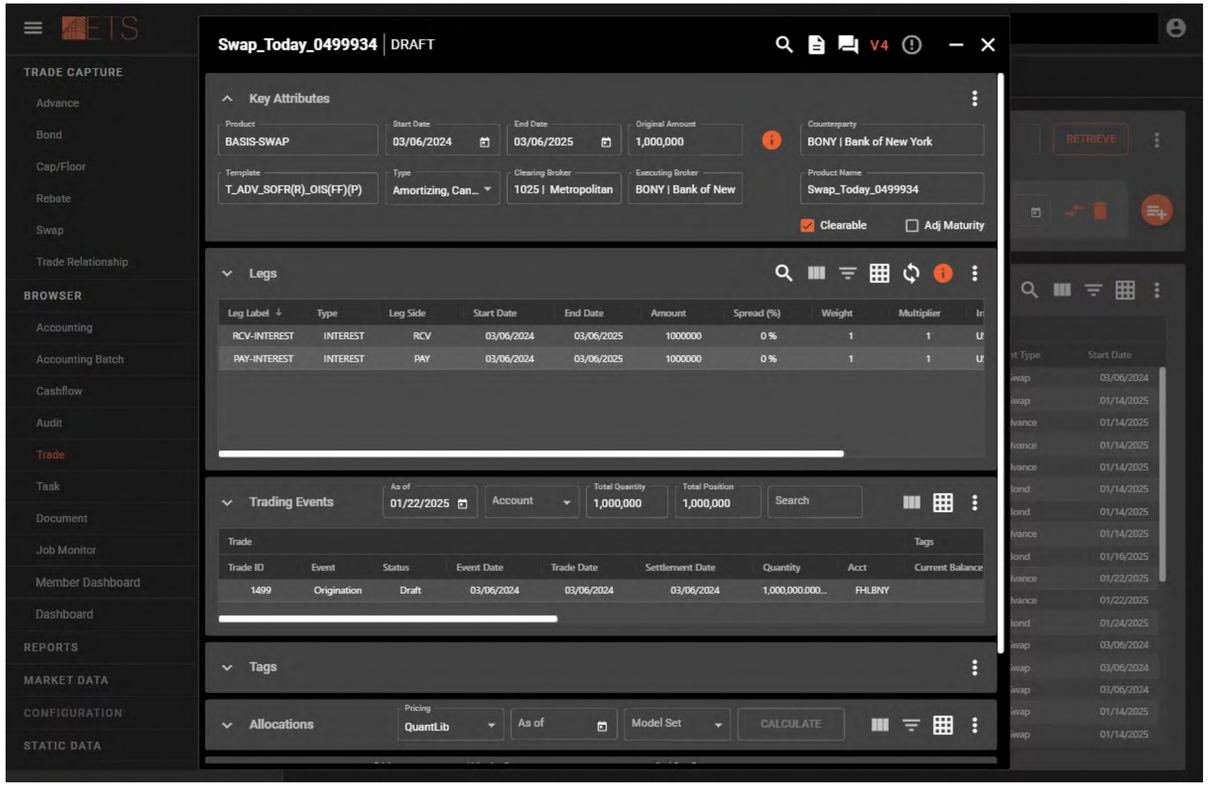
- The platform offers a **centralized, real-time view** of portfolios, with advanced rule engines, audit history, and flexible data management for institutional users.

- **Our team's mission**: Diagnose and resolve performance bottlenecks, improve system responsiveness, and enhance the scalability of the platform under heavy data and user load.

ForkJoin

Architecture: ETS is a **Java-based web application** using an **Oracle database** hosted on **AWS RDS**. The system is modular and deployed on AWS, using integrated monitoring tools.

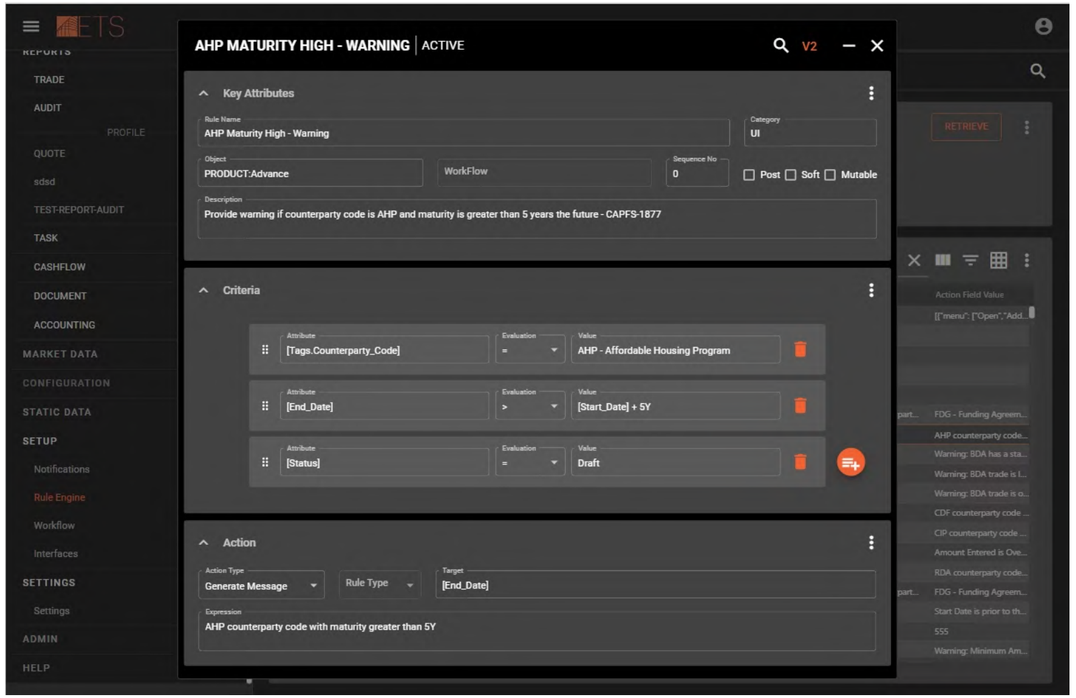


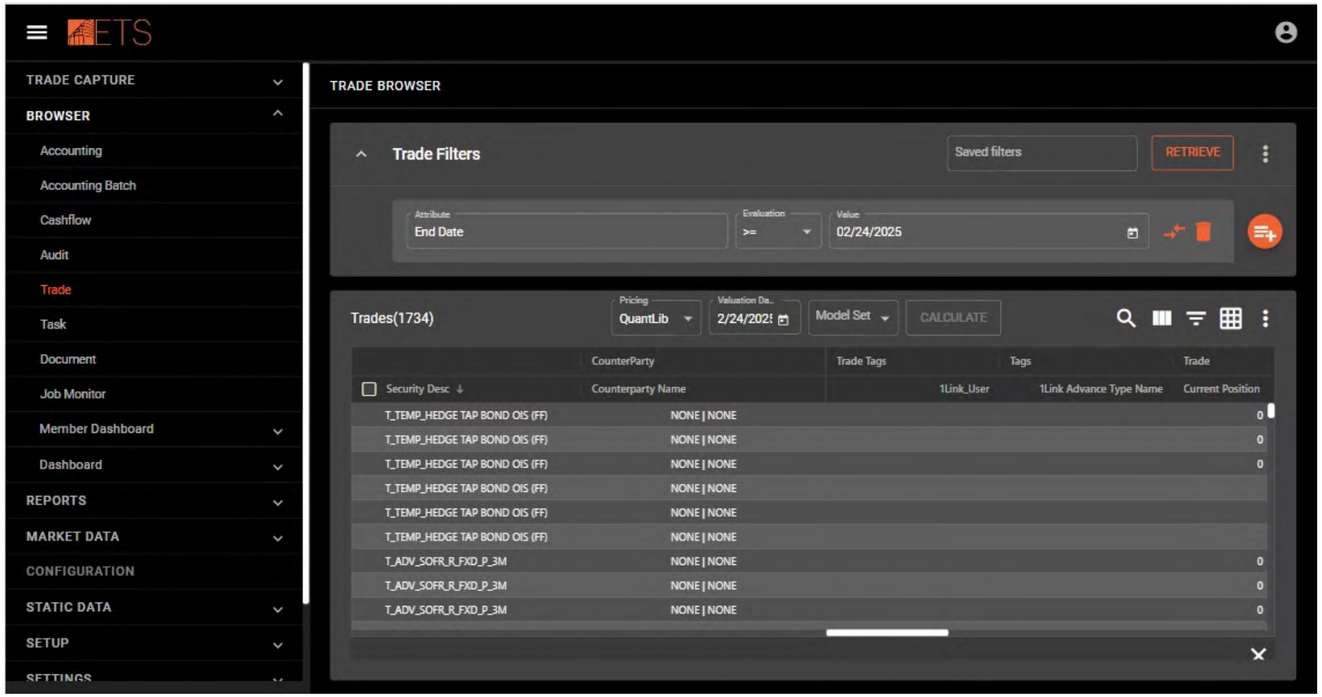


ForkJoin

Main UI Modules:

- **Member Dashboard** – A user-centric dashboard showing key stats and portfolio data. It offers quick access to workflows and tasks.
- **Trade Capture** – Allows entry and viewing of trades (bonds, swaps, loans). Includes validation and downstream integration.
- **Rule Engine** – A flexible rule builder for setting up workflows, validations, and automated compliance checks.





Key Issues Observed During Analysis:

- Slow audit queries: Fetching audit history for reference data tables (like domains or users) took 30–60+ seconds, often due to full table scans and lack of query optimization.
- Trade filtering performance: Trade search with multiple filters was slow (up to 40 seconds), especially for large datasets. Queries were not efficiently using indexes and relied on in-memory filtering.
- High infrastructure load: During peak usage, CPU on database servers reached 80–100%, and wait events caused cascading delays across user actions.
- Backend inefficiencies: Repeated queries, unnecessary data fetches, and non-optimized loops were consuming memory and CPU cycles.

ForkJoin

Optimization Techniques Applied

- SQL optimization: Added missing indexes, rewrote inefficient queries, and reduced table scan operations.
- Code refactoring: Eliminated N+1 query problems, simplified data processing logic, and deferred expensive computations (lazy loading).
- Pagination: Introduced paged queries for large result sets to improve performance and user experience.
- Caching & tuning: Implemented caching for static reference data and tuned the connection pool and JVM memory settings for better runtime efficiency.

Before vs After Results

Audit Performance:

Entity	Before (sec)	After (sec)	Improvement
Domain	~32.0	~1.3	~96% faster
Domain Value	~48.6	~1.3	~98% faster
Users	~66.7	~0.88	~99% faster
Counterparties	~3.2	~0.88	~73% faster
Roles	~1.51	~0.67	~56% faster

ForkJoin

Entity	Meaning / Description
Domain	A reference dictionary of data types, such as deal types, statuses, or classifications.
Domain Value	The specific values within a domain, e.g., if the domain is "Product Type", values might include "Loan", "Swap", "Bond".
User	Represents platform users. The audit logs show changes related to their roles, statuses, or account info.
Counterparty	Legal entities or clients involved in trades. The audit tracks edits to their profile data.
Role	Access control roles in the system. Audit shows who created or modified roles and permissions.

Before vs After Results

Trade Search Function:

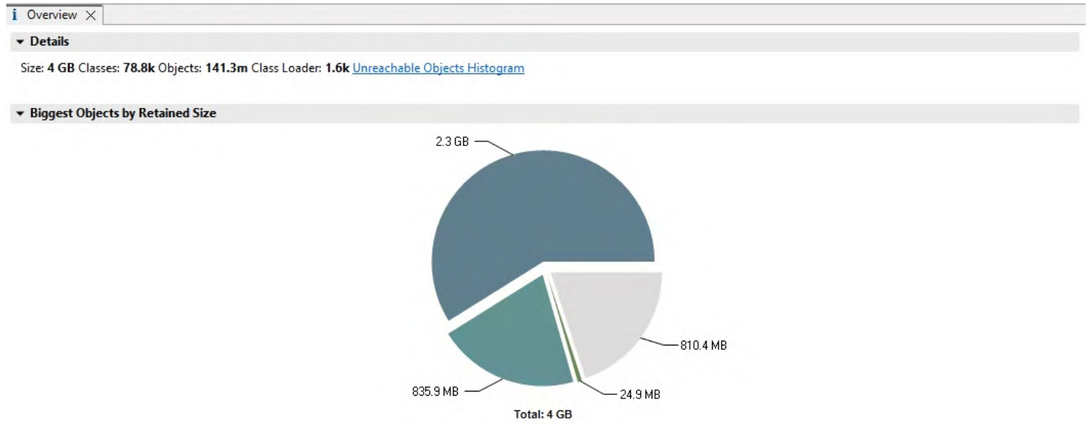
Dataset Size	Before (s)	After (s)	Improvement
659 trades	39.4	11.2	~72%
100 trades	7.7	3.6	~54%
33 trades	2.8	1.69	~40%

ForkJoin

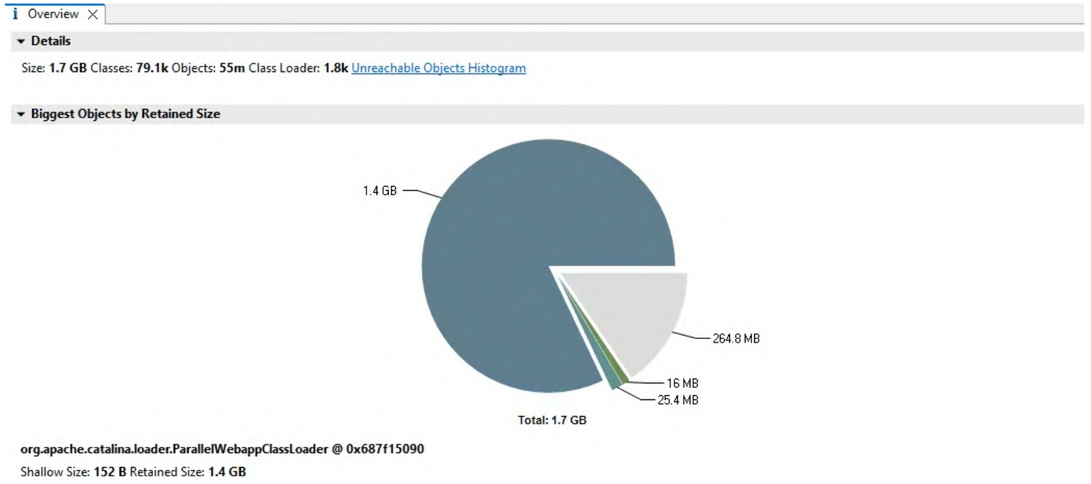
Infrastructure Metrics:

- **Database CPU Utilization:** Decreased from ~90% to ~35% during peak loads.
- **DB Load (AWS):** Lower active sessions, fewer lock waits, and improved query concurrency.
- **Memory & GC efficiency:** Optimized heap usage and faster garbage collection cycles, verified via profiling tools.

Before



After



Stability Enhancements

In addition to performance improvements, the system became significantly more **stable and resilient** under real-world usage. Several critical issues were identified and resolved:

Resolved critical DB slowdown under simultaneous large data requests:

Previously, if one or multiple users requested large volumes of data simultaneously, the system experienced extreme slowdowns at the database level, affecting all users. This issue was addressed by optimizing query execution, adding pagination, and limiting parallel heavy requests, configuring Oracle RDS to restrict resources amount for one heavy query to reduce its influence on the whole performance.

Fixed excessive CPU consumption during peak operations:

Certain backend operations caused sustained CPU usage spikes (up to 90–100%), resulting in overall system sluggishness. Refactoring these operations and reducing computational overhead significantly improved responsiveness.

Addressed high memory usage that risked application crashes:

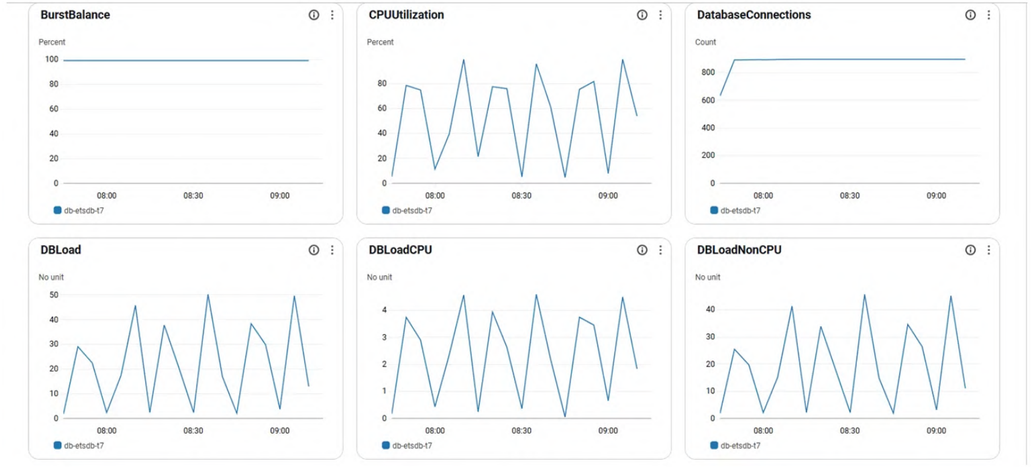
Some workflows led to memory-intensive processes that pushed the JVM close to critical limits. After identifying and eliminating memory leaks and optimizing object handling, memory consumption was stabilized and GC pressure was reduced.

ForkJoin

Key Takeaways

- **Major performance boosts:** Critical processes now execute **10–100x faster**.
- **Scalability enabled:** Lower CPU usage and query response time allow for handling larger data volumes and more users.
- **Cleaner architecture:** Backend improvements make future maintenance easier and reduce the chance of regressions.
- **Full-spectrum approach:** Using load testing + profiling + real-time monitoring delivered both short-term and long-term gains.

Cloud Watch



Performance Insights

Our Leadership Team

CEO

Michael Glazer

Michael has 25+ years experience in US Capital Markets. Over his career he held many different technical roles in Software Development, Solution Architecture and Delivery Management. Well-known key engagements included **FXall**, **Intercontinental Exchange** and **ENSO Financial** (Nex Group), and **GFT**.

Platonic meetperry NEX ENSO

ICE syniverse REFINITIV FXall GFT

CTO

Anton Moiseev

Anton has over 15 years experience with FinTech and Insurance projects, including leading insurance broker platform **SuranceBay**. Anton has a deep track record leading development teams and delivering production systems. Anton is also a Technical SME, contributing to open source projects, conducting webinars and co-authoring books.

Key open source creator for Dart.dev (later Flutter.dev)
Key architect and designer for https://www.surancebay.com/.
Notable **Manning Publications**:
TypeScript Quickly
Angular Development with TypeScript, Second Edition
Angular 2 Development with TypeScript

VP of Engineering

Michael Musatov

Michael is a Lead Development Manager with 15+ years of experience working on successful projects in Insurance, including leading insurance broker platform **SuranceBay**, FinTech and Telecommunications industries. Michael has strong skills in designing and implementing robust and simple solutions to solve complex real-life business cases with a rich variety of technologies including both Java and .NET platforms.

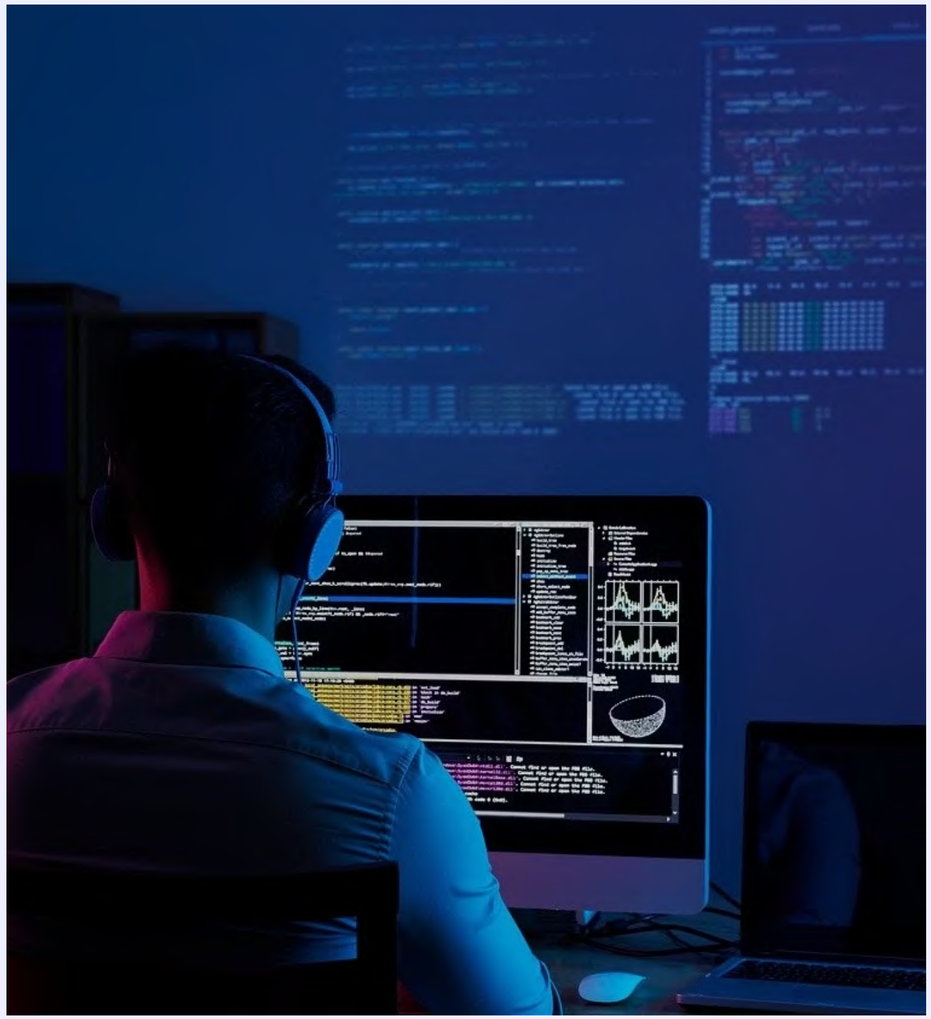
- MCPD: Windows Developer
- MCPD: Web Developer
- MCPD: Windows Azure Developer
- MCTS: .NET Framework, Web Applications
- MCTS: .Net Framework, Data Access
- MCTS: .NET Framework, Service Communication

Platonic

meetperry

SURANCEBAY
INSURANCE. SIMPLIFIED.

References



Capital Markets - ENSO Financial Analytics acquired by NEX (part of ICAP) and by Hazeltree

Functionality: Web Workflow, Reporting, Compliance, Market Data

Voted "Most Innovative Hedge Fund Platform" by Hedge Fund Magazine 2 years in a Row

Private Equity - Meetpery

Functionality: Web & Mobile Investor's Portal, Fund Transfer, Reporting

Assets Tokenization - Platonic (former Symbiont)

Functionality: Financial services with Platonic's enterprise-grade L1 blockchain.

Quantitative Trading - Epsilon Services

Functionality: Rich Real-Time Trading, Position Management

Telecom - REW - Tampa-based consulting firm focused on telecommunications

Functionality: Billing, SMS Notification, Customer Portals for large Users

ForkJoin

Contact Us

Website

www.forkjoin.io

Austria

(Home Office)

+43 676 3308473

New York

(Regional Office)

+1 (347) 420-1474

Email

hello@forkjoin.dev