Modulus Exchange High-Level Technical Overview

Introduction

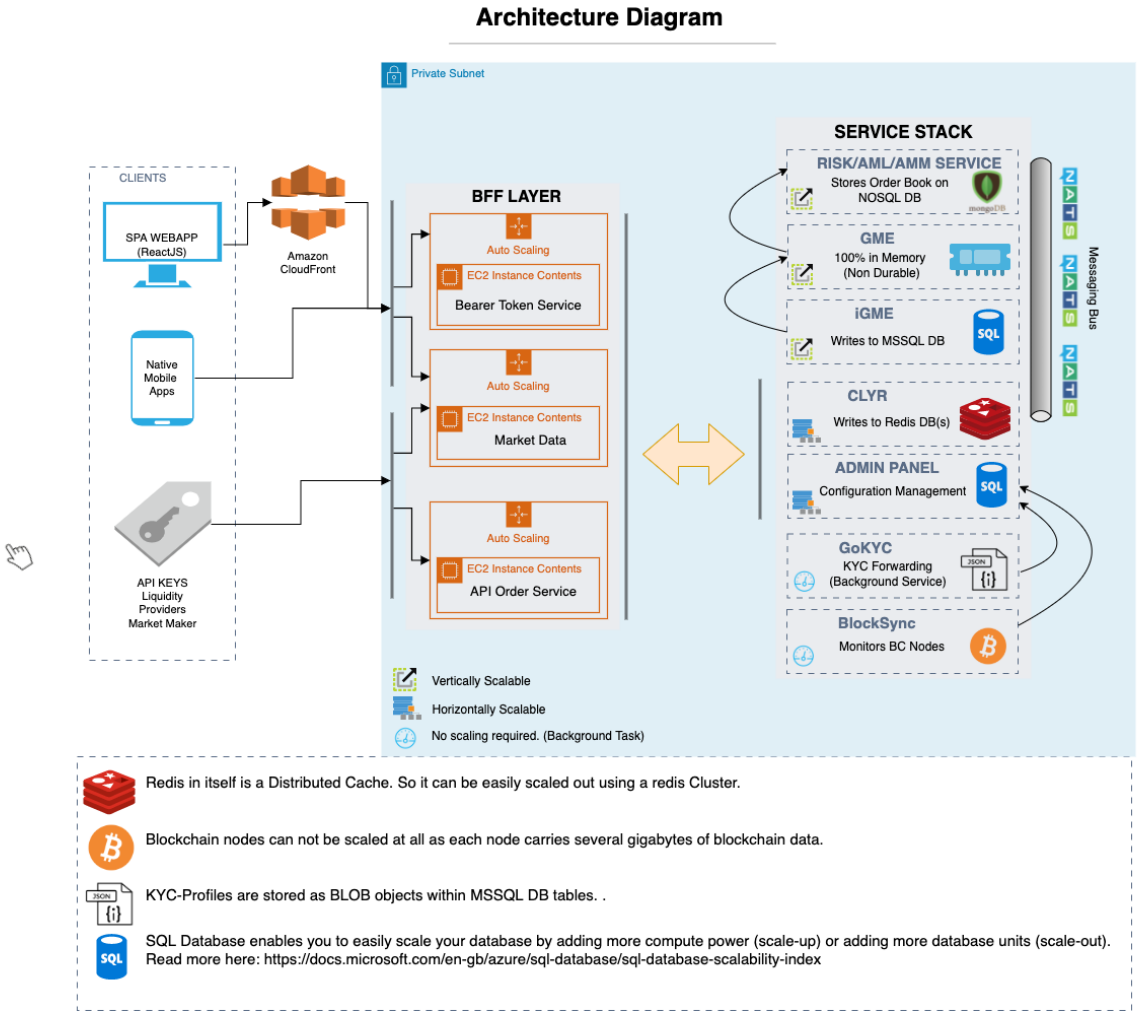
The Modulus Exchange Solution is a turnkey solution for block-chain based token trading. The solution has been purposefully designed with a microservice architecture. The blockchain components are executed in a secure server-side environment. The matching, risk, and trade surveillance engines have all been developed in Golang. The core business logic for block-chain manipulation has been developed in the C# programming language for rapid development and ease of customization.

Key Features

- 100% secure & closed proprietary code. No open source.
- All business logic resides on the server-side.
- 100% cold storage Integration for Ethereum and ERC20 tokens using 12-word mnemonic phrases. Keys never leave your computer.
- Official QT-Clients as a hot wallet for BTC, BCH, BTG, DOGE, DASH, LTC, USDT, and other Bitcoin forked coins. Private keys never leave the computer and are protected by having a wallet.dat file encrypted with a user-defined paraphrase.
- xPub based wallet integration available for deposit of popular coins like BTC, BCH, LTC, and others coming soon.
- Easy to list new coins and tokens. Bitcoin-forks and ERC20 Tokens listing take less than a minute.
- 84M order per second transaction processing speed.
- Easy to scale using MSSQL replication.
- Load balancer-friendly design using cross-device persistent session authentication and validation.
- Use of an enterprise-grade database, Microsoft SQL Server, that provides a consistent platform for secure and compliant modern data management needs. Provides many features like runs faster transactions with enhanced high availability and performance, Get in-database machine learning with R and Python as well as self-service reports and dashboards through Power BI Report Server and Mobile BI, Run mission-critical workloads using Windows, Linux, and containers on-premises, in the cloud, or in hybrid environments, Integrate, manage, and analyze both relational and unstructured big data using data virtualization and big data clusters.
- Built-in caching engine to cache order book, charting data, and more.
- Extensive use of ORM & LINQ (helps prevent SQL injection attacks).
- HTTP header obfuscation.
- Anti XSS code practices.
- Rest API (available), FIX API (future release).
- HMAC protection for replay attacks.
- Supports Google 2FA authentication.
- Supports SSL/TLS.
- Easy to customize colors and UI. Separate files for design & business logic.
- SignalR Websockets for real-time data streaming.
- Messaging Bus (NATS Streaming) for internal microservice communication.

- Referral Module available (referral code, referral link, referral reporting in user and admin panel).
- Fee discounting module available. Trade volume-based and pay-by-exchange token system.

System Architecture



Microservices

Service Name	OS	Publicly Accessible	Scalability
Admin	Ubuntu	Zone Locked	Not Required
API a.k.a BFF	Ubuntu	Yes	Horizontally
GME	Ubuntu	No	Both Ways
iGME	Ubuntu	No	Vertically
CLYR	Ubuntu	No	Vertically

Redis DB	Ubuntu	No	Clustered
NATS	Ubuntu	No	Clustered
MSSQL DB	Windows or Ubuntu	No	Clustered
BlockSync	Ubuntu	No	Not Required
Go-KYC	Ubuntu	No	Not Required
M3BOT	Ubuntu	No	Not Required

Admin

Admin panel is the core of all microservices and is comprised of customer profile management, customer wallets, deposits, withdrawals, trade engine, order management, risk engine, trade surveillance, charting, transactions reports, cold wallet integration, hot wallet integration, administrative reports built right into one package/assembly using .NET framework and for IIS 7 and above. One node of this layer can handle many admin users, so this layer doesn't really need scaling.

API Layer a.k.a BFF Layer

BFF shorts for the backend for frontend, this layer serves the ReactJS customer-facing frontend and serves most of the functionality that is publicly seen from the exchange website such as customer onboarding, orders, referrals, wallets, deposits, withdrawals, and KYC. This layer should be behind a load balance and must be scaled horizontally. On a typical server with 4 GB RAM & 2vCore CPU, this layer can handle about 1000 concurrent users.

GME

GME shorts for Golang matching engine, it's a pure in-memory service that can process up to 84 million transactions a second using 96 core processors on custom hardware (or 10 million TPS on Amazon EC2 T2.XLarge). An exchange can have a docker container for each product to scale out the matching engine.

iGME

iGME works in conjunction with GME and this layer takes the responsibility of starting the matching engine, restarting the matching engine, writing matches from matching engine to DB, recording charts to DB, and many other related functions. This layer cannot be scaled out horizontally as it is a singleton service and must be scaled vertically.

CLYR

CLYR shorts for caching layers and like iGME this layer works in conjunction with GME to take the responsibility of writing order book cache, user open order cache, and other statistics to the Redis DB. This layer cannot be scaled out horizontally as it is a singleton service and must be scaled vertically.

BlockSync a.k.a Deposit Sync service (MS2)

This service has the responsibility of monitoring various blockchains and listens for incoming payments. This service keeps a local copy of all addresses it has given to the DB service and intimates DB service soon after a transaction on a blockchain receives the desired number of confirmations. The prime function is to deliver real-time deposit

notifications and generate email notifications to clients. Being a cron job type of service, this service is not resource-intensive and thus doesn't need scaling.

NATS Jetstream

NATS Streaming is a data streaming system powered by NATS and written in the Go programming language. We use NATS primarily to communicate between microservices. NATS can run as a cluster, read more here: https://docs.nats.io/nats-server/configuration/clustering#:~:text=NATS%20Server%20Clustering,dynamically%20forming%20a%20full%20mesh.

Redis DB

Redis being a fast in-memory DB offers the capability to store common data that all microservices need in addition to the distributed wallet locks and pub/sub channels that BFF uses to communicate with other BFF nodes. Redis can run as a cluster, read more here: https://redis.io/topics/cluster-tutorial#:~:text=Redis%20Cluster%20provides%20a%20way,are%20not%20able%20to%20communicate.

MSSQL DB (Persistent Storage)

Microsoft SQL Server 2019 Standard Edition is used for database storage and can be scaled out & clustered easily, read more here: https://www.mssqltips.com/sqlservertip/4879/sql-server-vnext-sql-server-2017-ssis-scale-out-feature/

Go-KYC

Go-KYC is kind of a hub service that connects with various KYC providers such as Trulioo, IdentityMind, Sum & Substance, SynapseFi, ShuftiPro, and others. With this module, you can pass identity documents to the backend service providers.

M3BOT

M3BOT is a lightweight market-making Bot that can copy order books from many exchanges and brings liquidity to the exchange. It has a Straight order routing engine built-in, that takes the responsibility of settlement of matches on the source exchange as soon as a customer trades with the Bot. As it's a lightweight and stateless service, it doesn't need scaling.

Frontend

The web frontend is written in React and is built to be responsive and performant. Code dependencies in the frontend project are kept to a minimum to make it easy for customers to extend the frontend themselves.

Mobile Apps

The mobile applications are written in native Swift (iOS) and Java (Android).

Blockchain Nodes

We use official blockchain nodes commonly known as core wallets or QT clients. Typically, all Bitcoin & Bitcoin forked coins have QT clients for Windows, Linux, and Macintosh. QT clients can be hosted anywhere and on any OS.

For deployment and/or orchestration notes, please check the following article:
https://modulushelp.freshdesk.com/support/solutions/articles/43000630227-microservices-orchestration

Knowledge Base

Clients receive access to our knowledge base.