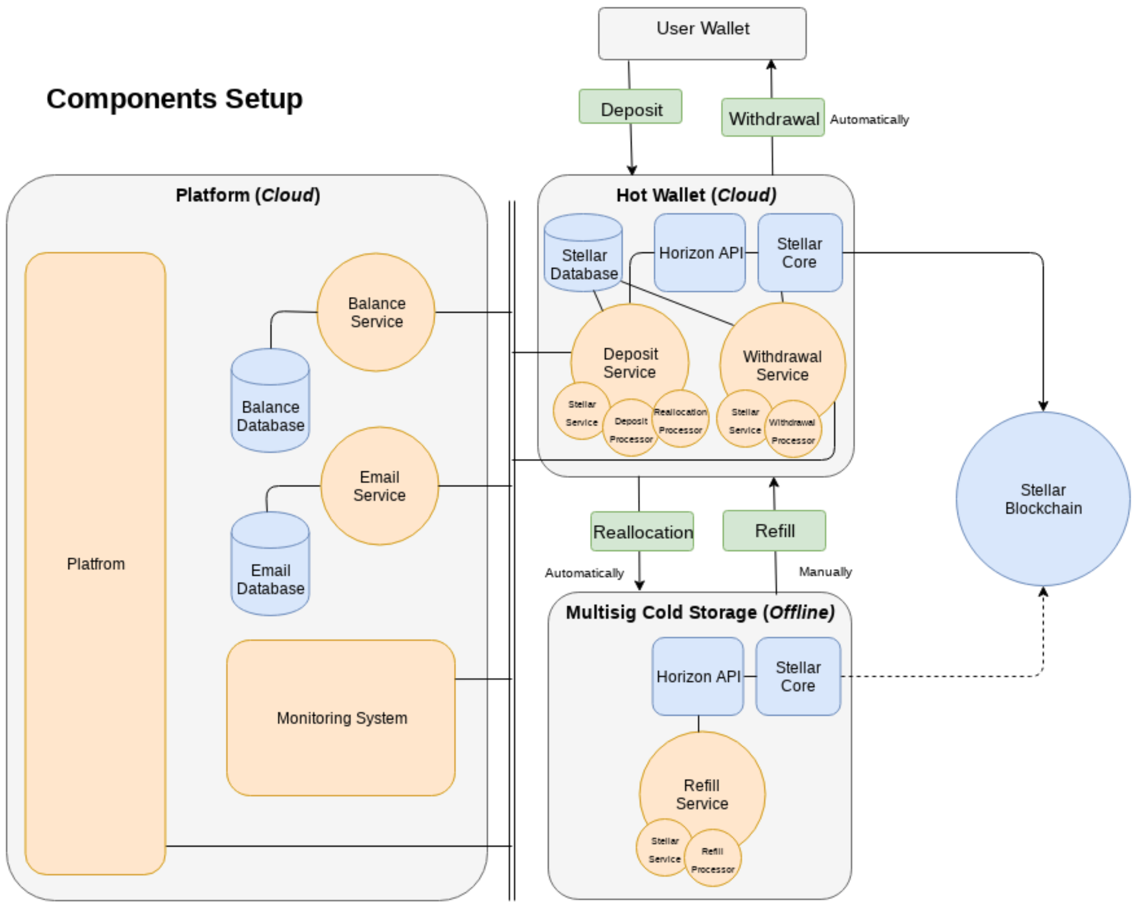
Components Setup



For Stellar Integration Cold and Hot storage should be setup.

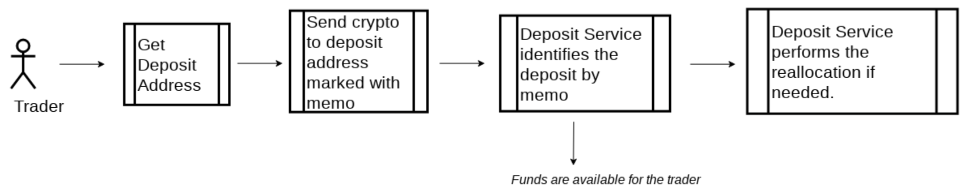
Hot Storage consists of Stellar Core component and Horizon API server, additionally two microservices work in cloud - Deposit service and Withdrawal service. We setup deposit account with simple single key. This account will be receiving deposit payments from traders. We will distinguish deposits by memo value, which is attached to transaction that contains payment operation. Whenever Trader wishes to deposit XLM to his account at Traderio, he requests for deposit address from Platform, which we provide to him together with generated CustomerId. This unique CustomerId trader should attach to transaction which will be sent to deposit address. We need to draw users attention to the necessity of attaching CustomerId in memo field of the transaction, in other case we will not be able to identify trader and so funds will be irreversibly lost. Internally we keep Trader integer identifier which is used to store trader balances in the system, record fees, facilite trades and so on. When user requested for XLM deposit address we generated unique CustomerId and stored it for this trader in Stellar Database. This CustomerId should be generated only once when first requested, later on we will be returning existing CustomerId. So when Deposit service listens for incoming payments, we extract memo field from transaction that cointanes received payment, and try to find trader with CustomerId equal to extracted memo. If trader is identified we deposit payment amount to trader XLM balance and record transaction in Stellar database. If trader was not identified Deposit service will send email notification with details, so customer support may decided how to resolve this.

As we are getting more funds on deposit account we need to move exceeding funds to cold storage, we will call this operation reallocation. We need to decide what will be the threshold when reallocation should be done. We will consider this value as 10% of all funds of XLM. Reallocation will be made automatically by Deposit service, inside there will be reallocation processor that will hold this logic for making decision and submitting signed transaction. We will be using deposit account also to process withdrawal requests from traders. For reallocation when calculating exceeding funds, amount of pending withdrawals should be deducted from hot wallet funds. This will prevent the situation when hot wallet does not have enough funds to process withdrawal request.

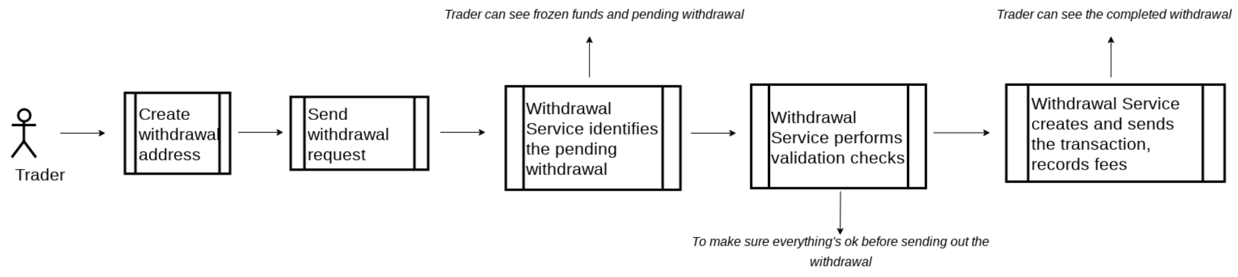
When trader wishes to withdraw XLM funds from platform he will provide withdrawal address together with amount. This request is stored in Stellar database. Withdrawal service will process pending withdrawal requests. It should check whether trader balance is sufficient and whether the hot wallet balance is also sufficient. If former is false there will email notification with details, in latter - there will be email notification with refill warning. Refill transaction should be made from cold wallet to hot wallet.

We will setup multisig cold wallet account with 6 keys, but only 3 keys will be needed to make the refill transaction. We can partially automate creation of refill transaction in sense that when trusted users provide their secrets refill service constructs and signes transaction. Then we temporarily connect to the web to submit transaction to blockchain.

Deposit Flow


Trader
Get Deposit Address
Send crypto to deposit address marked with memo
Deposit Service identifies the deposit by memo
Deposit Service performs the reallocation if needed.
Funds are available for the trader

Withdrawal Flow


Trader can see frozen funds and pending withdrawal
Trader can see the completed withdrawal
Trader
Create withdrawal address
Send withdrawal request
Withdrawal Service identifies the pending withdrawal
Withdrawal Service performs validation checks
Withdrawal Service creates and sends the transaction, records fees
To make sure everything's ok before sending out the withdrawal

Solution structure.

1.Tradeio.Stellar.Deposit – microservice with Stellar Deposit API endpoints (requesting deposit address, getting deposits, getting service status for monitoring), it holds singleton DepositProcessor and ReallocationProcessor. Keeps OpenApi specification for microservice, and nsag studio project file to automatically generate microsirvice controlers and client classes to consume stellar depost api.

2.Tradeio.Stellar.Withdrawal – microsorvice with Setller Withdrawal API endpoints (submitting withdrawal request,Getting withdrawals, getting service stauts for monitoring system), it hold singleton WithdrawalProcessor. Keeps OpenApi specification for microservice, and nsag studio project file to automatically generate microsirvice controlers and client classes to consume stellar depost api.

3.Tradeio.Stellar.Data – data models, ef core context for database, database migrations, stellar repositiory.

4.Tradeio.Stellar.Configuration – settings classes and service to be injected in all needed components.

5.Tradeio.Stellar.Client – client classes to consume stellar deposit and withdrawal services api.

6.Tradeio.Stellar – contains main logic to process deposit, withdrawal and reallocation flows. Also contains stellar service component, which encupsulates functionality to deal with Horizon API, it uses .NET Stellar SDK nuget package to query data and to build and sign transactions.

7.Tradeio.Stellar.Tests – test project for testing domain logic for deposit, withdrawal and reallocation processing, and to test stellar service for main operations such as query account balance, query transaction, submitting transaction.

8.Tradeio.Email – prototipy for email service component encapsulating logic for sending emails with notification such as, insuficient trader balance, insufficient hot wallet balance, relloaction failure, unregistered trader deposit.

9.Tradeio.Balance – prototype for balance service encapsulating logic to manage traders balances, it is used to request trader balance and to make trader balance changes such as deposit and withdrawals.