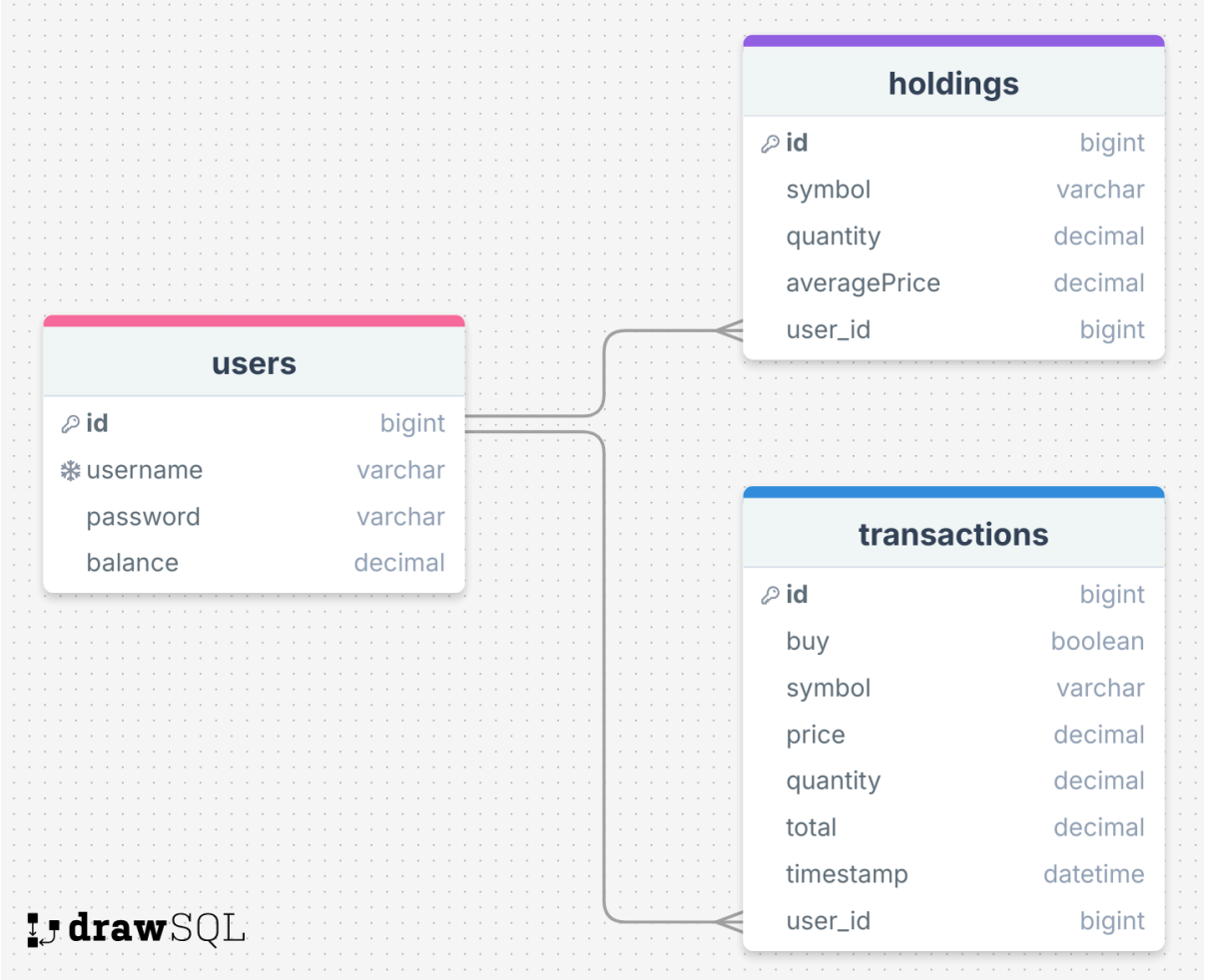
Crypto Trading Simulator

By Evgeni Nikolov

Design and implementation

Database overview - MySQL



Backend Implementation

1. Entities
    a. User
    b. Transaction
    c. Holding

2. Repositories
    a. UserRepository - Repository class for managing user entities in the database. Provides methods for database operations related to users.

    b. TransactionRepository - Repository class for managing transaction entities in the database. Provides methods for database operations related to transactions.
    c. HoldingRepository - Repository class for managing holding entities in the database. Provides methods for database operations related to holdings.

3. Services
    a. UserService - Service class for user-related logic. Handles user registration and delegates database operations to the UserRepository.
    b. TransactionService - Service class for handling transaction-related logic. It interacts with the TransactionRepository to perform database operations related to transactions.
    c. HoldingService - Service class for managing holdings. It provides methods to save or update holdings, retrieve holdings by user and symbol, and retrieve the average price of holdings.
    d. TradeService - Service class for handling trade-related logic. Manages buy and sell transactions, updates user balances, and maintains holdings.
    e. CoinSyncService - Service for synchronizing and caching the top 20 cryptocurrencies between Kraken API and CoinGecko API
4. Controllers
    a. UserController- REST controller for user-related operations.
    b. TradeController - REST controller for trade-related operations.
    c. CoinController - REST controller for handling coin-related operations.
5. DTO
    a. CoinDTO - Data Transfer Object for coin information. Used for fetching and displaying coin information.

API Endpoints

1. User (/api/users)
    a. (/register) - POST endpoint which registers a new user
    b. (/login) - POST endpoint which returns a user if found
    c. (/holdings) - POST endpoint which returns user's holdings
    d. (/averagePrice) - POST endpoint which returns the averagePrice of user's holding based on specific coin
    e. (/transactions) - POST endpoint which returns the user's transactions
    f. (/{id}) - GET endpoint used for retrieving the user by their ID
    g. (/{id}/reset - POST endpoint used for resetting the user's account balance, their holdings and transactions

2. Trade (/api/trade)
    a. (/buy) - POST endpoint to handle buy requests
    b. (/sell) - POST endpoint to handle sell requests

3. Coin (/api/coins)
    a. (/top20) - GET endpoint to retrieve the top 20 coins

Challenges

Retrieving the top 20 crypto coins:

The requirements of this project stated that the Kraken V2 Websocket API should be used, and that top 20 coins should be displayed with real time price updates. However this API does not provide the market cap of the coins or any sorting endpoints. It also doesn't provide names for the coins. In order to get around this the top 20 coins are fetched from the CoinGecko API every 15 minutes (to avoid reaching api call limits). The fetched coins are then synced with coins available on the Kraken API. A list of CoinDTOs is created. Certain coins on the Kraken API use different symbols than what is widely used, so an Alias map was required to correctly sync certain popular coins such as Bitcoin and Doge. The Kraken Websocket API is then used to fetch the realtime prices of coins.

Managing floating point errors

There were floating point issues, particularly when trying to use up the entire account balance (e.g. could get the account into the negatives). To tackle this an epsilon value was created to act as a small threshold when comparing BigDecimal (float) values.