CS5040A: Database Systems -- Graduate

Database Design and Development Project

Application Proposal

***Title*:**

User Financial Management System - Trading Application

***Purpose*:**

Managing and storing different User Information for trading application and Manage user account and Maintain up to date about financial data.

***Description*:**

Financial trading database enables the financial data information about user profiles , Market Pricing , Trade requests , Account and Transaction information , Product information. Managing the User portfolio and account details and Up to date with Market performance ensures user to analyze better way to invest and get profits. The Main Purpose of this Database is to Manage and store user information about individuals investments and analyze the data for different use cases in terms of trade analysis which helps users about their management of stocks and assets.

***User Groups*:**

(1) Users

(2) Accounts

(3) Orders

(4) Instrument(Product)

(5) Wallets

(6) Roles(Portfolio)

(7) Transaction

(8) Market Info

***Usage by User Group*:**

(1) Users

- Users group stores the information about different parameters of User personal information
- It helps to store personal information like UserName , Password , Email , CreatedAt
- It helps user to create their Username and Password and store in backend in secure way
- At the time of creation it will store the registration information about user for data tracking and second level verification is required (2FA)
- User has multiple user groups having different group members can trade and discuss

(2) Accounts

- It stores account_id information about specific user_id (individual)
- User can add multiple account details for deposit / withdraw amount of investment which can reflect that data in wallet.
- It Can show how much balance information is currently have based on account_type and refresh for 1ms Every time
- For showing balance it has various currencies dependency of user needs , user can check the balance

(3) Orders

- Every order has Order_id which stores order_type of stocks (Buy/Sell)
- Order has instrument ID / product ID which stores instrument/product information
- Order_type helps to find that user will buy/sell with when user bought / sold that stock with status
- Order details have price information and how much quantity that user has bought for record base

(4) Instrument

- It helps to find information about what type has user (stock/crypto/ forex etc) with respective symbol
- It will map with exchange data information which helps the exchange data (currency and symbol) and country
- It has watchlist items of users and check watchlist data and list options that to explore

(5) Wallets

- User has wallet with two options
- Withdraws and deposits
- Condition both have amount log stored with timestamp with currency

(6) Roles

- User has role it might be either broker , end user
- Can store user roles with role data information and intersect with Users.

(7) Transaction

- Every Order has transaction data of values with time stamp
- Transaction history store transaction time and date with status (success/failure)

(8) Market Data

- Market data have entire details of opening and closing price with high price and percentage of increase with timestamp

It includes different settings options , notification messages to user device and store user session and ensure about location activity.

CS5040A: Database Systems -- Graduate

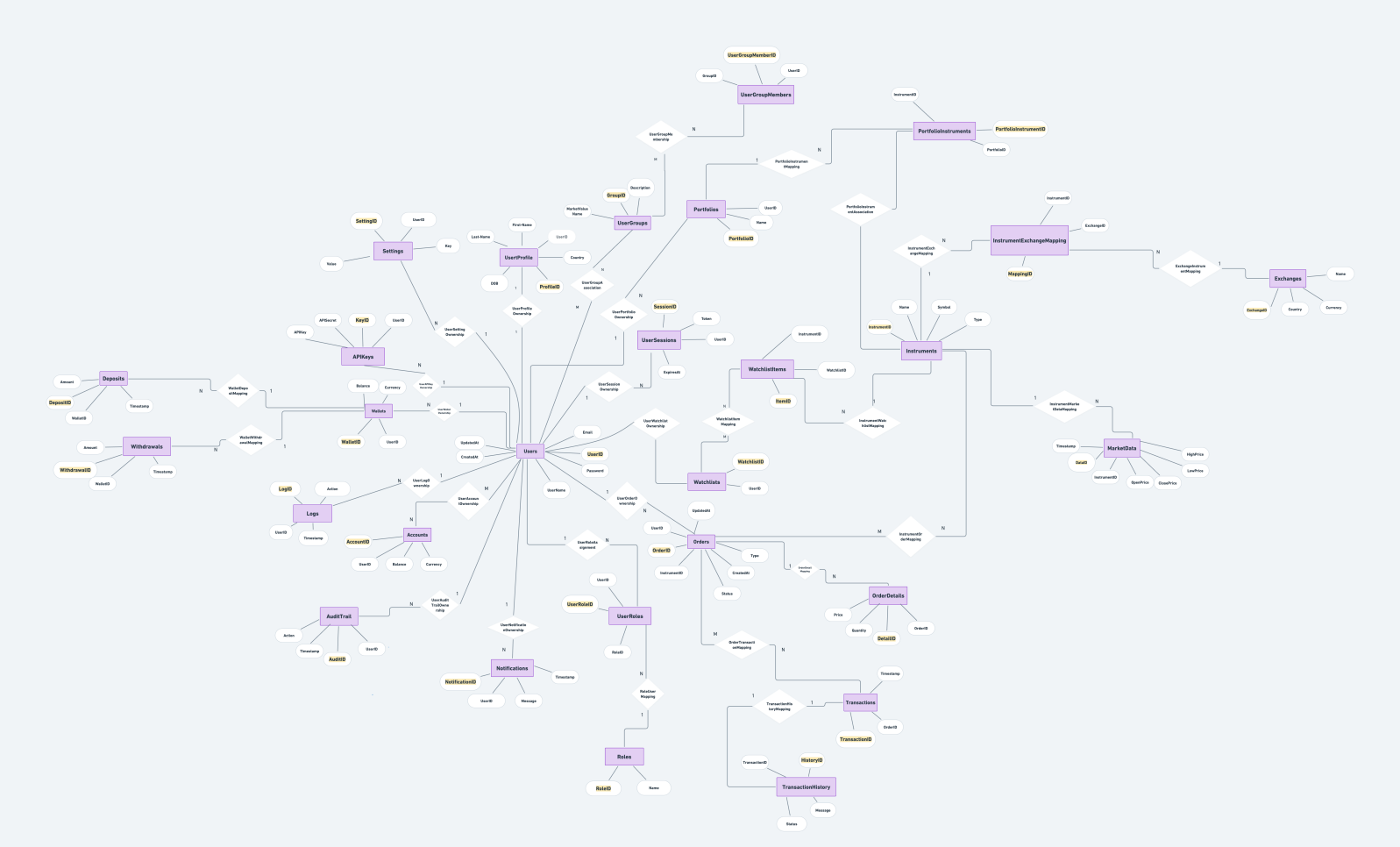
***Business Rules*:**

- Every user has their individual information like Name , Contact info , Email etc
- Along side the user has their Username and password respectively
- At the time of user register first time , it contains registration information for user log tracking for history of his account
- Account has min deposit of $10 and max of $250000 and same with withdrawal limit in wallets
- Can show the balance on daily basis and it will update once market period completed
- Every company has specific company symbol and organization name and market capital set with stock price of share is also different based on company value
- Orders buy/sell By user at any time and can do vice versa based on individual decision , Only restriction that user will do during market period
- Order can store purchase log for user at that time for user history of trading and user have product (stock) and can buy min 1 company
- Each user has own role based and account management capability and can provide access to other user with duplicate account(if we can create we can do that)
- Every transaction needs to be store which will perform by user / individual and can store in transaction_history
- Notification messages alert based on condition of trade
- Settings option to enable and audit trail to store session information and trade information

Note :Basic EER Diagram has designed and based on application with parameters we can modify the database based on course relate

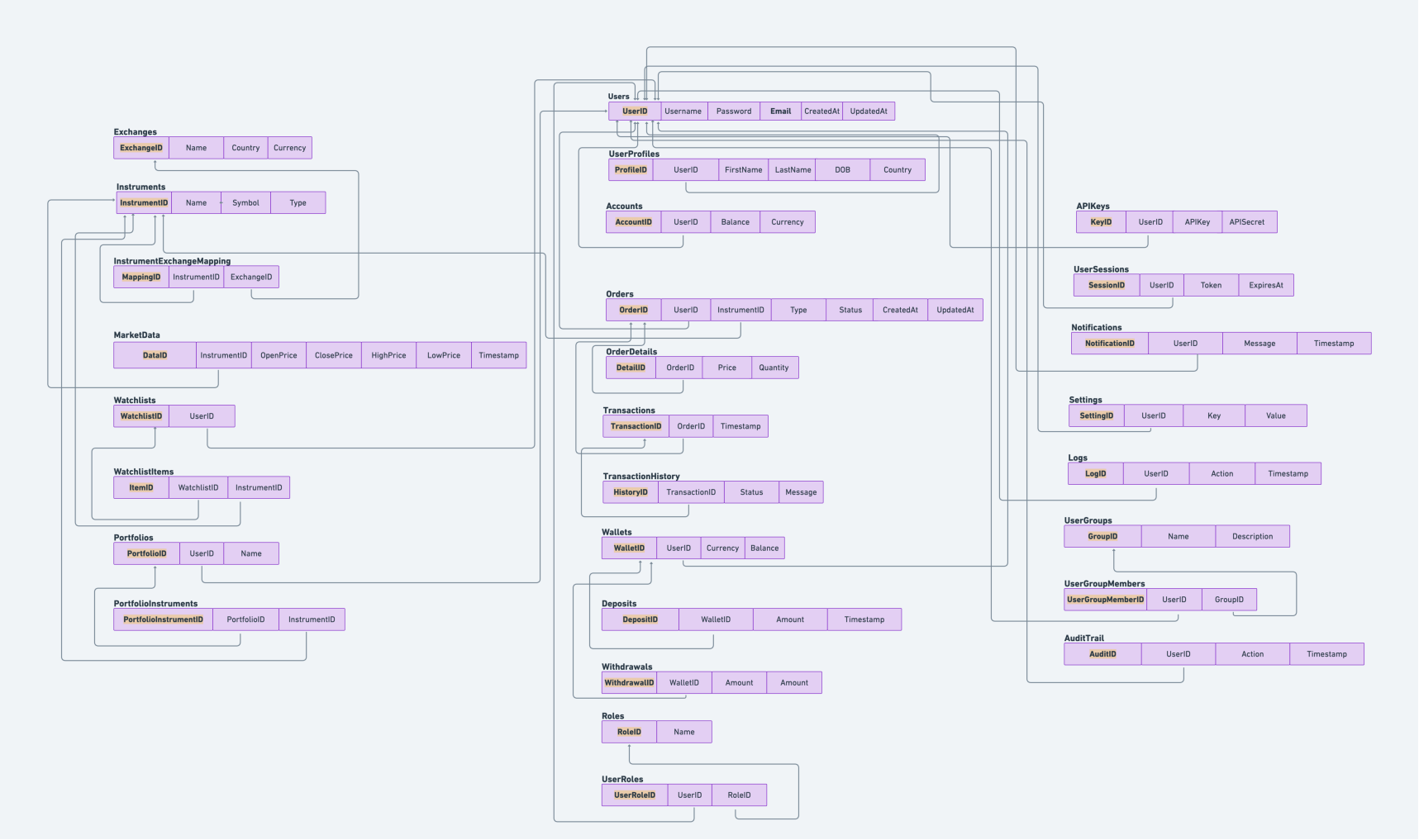
ER Schema Diagram:

Link for ER schema - https://whimsical.com/step1-XUaEk76s1mdahPtQ6cheBP



Relation Schema :

Link for Relation Schema - https://whimsical.com/step-2-FPYcXFnpB6E12gGqk8onYa



In addition to that here are the additional constraints implicated to each table, excluding primary key and foreign key constraints:

Users
- Username UNIQUE
- Email UNIQUE
- CreatedAt and UpdatedAt TIMESTAMP

UserProfiles
- FirstName, LastName VARCHAR(50)
- DOB DATE
- Country ENUM('USA', 'UK', 'Canada', ...)

Accounts
- Balance DECIMAL(15, 2)
- Currency ENUM('USD', 'EUR', 'GBP')

Instruments
- Name VARCHAR(100)
- Symbol UNIQUE
- Type ENUM('Stock', 'ETF', 'Crypto')
- **Exchanges**
- Name UNIQUE
- Country VARCHAR(50)
- Currency ENUM('USD', 'EUR', 'GBP')

InstrumentExchangeMapping
- MappingID UNIQUE

Orders
- Type ENUM('Buy', 'Sell')
- Status ENUM('Open', 'Closed', 'Cancelled')
- CreatedAt, UpdatedAt TIMESTAMP

OrderDetails
- Price DECIMAL(10, 2)
- Quantity INT

Transactions
- Timestamp TIMESTAMP

TransactionHistory
- Status ENUM('Success', 'Failure')
- Message TEXT

Wallets

- Currency ENUM('USD', 'EUR', 'BTC')
- Balance DECIMAL(15, 2)

Deposits
- Amount DECIMAL(10, 2)
- Timestamp TIMESTAMP

Withdrawals
- Amount DECIMAL(10, 2)
- Timestamp TIMESTAMP

APIKeys
- APIKey UNIQUE
- APISecret UNIQUE

UserSessions
- Token UNIQUE
- ExpiresAt TIMESTAMP

Watchlists
- No additional constraints

WatchlistItems
- ItemID UNIQUE

Notifications
- Message TEXT
- Timestamp TIMESTAMP

Roles
- Name ENUM('Admin', 'User', 'Guest')

UserRoles
- No additional constraints

Logs
- Action TEXT
- Timestamp TIMESTAMP

Settings
- Key VARCHAR(50)
- Value TEXT

UserGroups
- Name UNIQUE
- Description TEXT

UserGroupMembers
- No additional constraints

Portfolios
- Name VARCHAR(100)

PortfolioInstruments
- No additional constraints

Below are some Complex Constraints for tables
- **Accounts**: A user cannot have more than one account with the same currency.
- **WatchlistItems**: A watchlist cannot have duplicate instruments.
- **UserRoles**: A user cannot have duplicate roles.
- **UserGroupMembers**: A user cannot belong to the same group more than once.
- **PortfolioInstruments**: A portfolio cannot have duplicate instruments.

These constraints include domain constraints for attributes and more complex intra or inter-relation constraints.