# **TradePulse**

# **(CRYPTO-TRADING-PROJECT )**

# ER model -

https://almondine-countess-b08.notion.site/TradePulse-Fullstack-Trading-Project-React-Java-d635590f7e3148cda6a081da96f7b2f2?pvs=4

+---------------------+ +-----------------+

| Users |<--------->| Wallets |

|---------------------| +-----------------+

| id | ^

| fullName | |

| email | |

| ... | |

+---------------------+ |

|

+--------------------+ +-----------------+

| Assets |<---------->| WalletTransactions |

|--------------------| +-----------------+

| id |

| quantity |

| buy\_price |<---------->+-----------------+

| coin\_id | | Coins |

| user\_id | +-----------------+

+--------------------+ | id |

| symbol |

+--------------------+ | ... |

| Withdrawals |<---------->+-----------------+

|--------------------|

| id |

| status |

| amount |

| user\_id |

| date |

+--------------------+

+--------------------+

| Watchlists |

|--------------------+

| id |

| user\_id |

+--------------------+

|

|

v

+--------------------+

| Watchlist\_Coins |

|--------------------+

| watchlist\_id |

| coin\_id |

+--------------------+

+---------------------+ +---------------------+

| VerificationCodes |<--------->| Users |

|---------------------| +---------------------+

| id |

| otp |

| user\_id |

| email |

| mobile |

| verification\_type |

+---------------------+

+---------------------+ +---------------------+

| TradingHistories |<--------->| Users |

|---------------------| +---------------------+

| id |

| selling\_price |

| buying\_price |

| coin\_id |

| user\_id |

+---------------------+

+---------------------+ +---------------------+

| PaymentOrders |<--------->| Users |

|---------------------| +---------------------+

| id |

| amount |

| status |

| payment\_method |

| user\_id |

+---------------------+

+---------------------+ +---------------------+

| PaymentDetails |<--------->| Users |

|---------------------| +---------------------+

| id |

| account\_number |

| account\_holder\_name |

| ifsc |

| bank\_name |

| user\_id |

+---------------------+

+---------------------+ +---------------------+

| Orders |<--------->| Users |

|---------------------| +---------------------+

| id |

| user\_id |

| order\_type |

| price |

| timestamp |

| status |

| order\_item\_id |

+---------------------+

|

|

v

+---------------------+ +---------------------+

| OrderItems |<--------->| Coins |

|---------------------| +---------------------+

| id |

| quantity |

| coin\_id |

| buy\_price |

| sell\_price |

| order\_id |

+---------------------+

+---------------------+ +---------------------+

| Notifications | <---------> | Users |

|---------------------| +---------------------+

| id |

| from\_user\_id |

| to\_user\_id |

| amount |

| message |

+---------------------+

+---------------------+

| MarketChartData |

|---------------------|

| id |

| timestamp |

| price |

+---------------------+

+---------------------+ +---------------------+

| ForgotPasswordTokens|<--------->| Users |

|---------------------| +---------------------+

| id |

| user\_id |

| otp |

| verification\_type |

| send\_to |

+---------------------

This is a detailed Entity-Relationship (ER) model that represents a system for managing a cryptocurrency trading or investment platform. Here's an explanation of the key entities and their relationships:

**1. Users**

* Represents the users of the system.
* Attributes:
  + id: Unique identifier for a user.
  + fullName, email, and other attributes define user information.
* Relationships:
  + Associated with **Wallets**, **VerificationCodes**, **TradingHistories**, **PaymentOrders**, **PaymentDetails**, **Orders**, **Notifications**, and **ForgotPasswordTokens**.

**2. Wallets**

* Represents a user's cryptocurrency wallet.
* Relationships:
  + Connected to **Users** (a user has a wallet).
  + Linked to **WalletTransactions** for recording transactions.

**3. WalletTransactions**

* Tracks wallet transactions.
* Attributes:
  + References **Wallets** and **Coins** for transaction details.

**4. Assets**

* Represents the cryptocurrency assets owned by a user.
* Attributes:
  + id, quantity, buy\_price, coin\_id, user\_id.
* Relationships:
  + Links to **Coins** and **Users** to determine ownership and coin type.

**5. Coins**

* Represents different cryptocurrencies (e.g., Bitcoin, Ethereum).
* Attributes:
  + id, symbol, and other attributes to define coin details.

**6. Withdrawals**

* Tracks withdrawal transactions.
* Attributes:
  + id, status, amount, user\_id, date.
* Relationships:
  + Associated with **Users**.

**7. Watchlists and Watchlist\_Coins**

* **Watchlists** allow users to monitor selected cryptocurrencies.
* Attributes:
  + Watchlists: id, user\_id.
  + Watchlist\_Coins: watchlist\_id, coin\_id.

**8. VerificationCodes**

* Stores verification details for user authentication.
* Attributes:
  + otp, email, mobile, verification\_type, etc.
* Relationships:
  + Linked to **Users**.

**9. TradingHistories**

* Maintains a log of user trades.
* Attributes:
  + selling\_price, buying\_price, coin\_id, user\_id.
* Relationships:
  + Connected to **Users** and **Coins**.

**10. PaymentOrders and PaymentDetails**

* **PaymentOrders**:
  + Tracks user payment transactions.
  + Attributes: amount, status, payment\_method, etc.
* **PaymentDetails**:
  + Stores user bank information.
  + Attributes: account\_number, ifsc, bank\_name, etc.
* Relationships:
  + Both are linked to **Users**.

**11. Orders and OrderItems**

* **Orders**:
  + Tracks user orders (buy/sell).
  + Attributes: order\_type, price, timestamp, etc.
  + Linked to **OrderItems** and **Users**.
* **OrderItems**:
  + Defines individual items within an order.
  + Attributes: quantity, coin\_id, buy\_price, sell\_price.
  + Connected to **Coins**.

**12. Notifications**

* Manages user notifications.
* Attributes:
  + from\_user\_id, to\_user\_id, amount, message.
* Relationships:
  + Linked to **Users**.

**13. MarketChartData**

* Stores cryptocurrency market data.
* Attributes:
  + timestamp, price.

**14. ForgotPasswordTokens**

* Handles password recovery.
* Attributes:
  + otp, verification\_type, send\_to.
* Relationships:
  + Associated with **Users**.

**Summary of Relationships:**

* **Users** are central to the system, interacting with almost every other entity.
* The system tracks wallet activities, orders, payments, trading histories, and watchlists.
* Coins play a key role in connecting entities like assets, orders, and trading histories.

.

Here’s a breakdown of the relationships between entities in terms of **one-to-one**, **one-to-many**, and **many-to-many**:

**1. Users and Wallets**

* **Relationship:** **One-to-One**
* **Explanation:** Each user has one wallet, and each wallet is associated with only one user.

**2. Wallets and WalletTransactions**

* **Relationship:** **One-to-Many**
* **Explanation:** One wallet can have many transactions (deposits, withdrawals, etc.), but each transaction belongs to a single wallet.

**3. Users and Assets**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can own many assets (cryptocurrencies), but each asset is linked to one user.

**4. Assets and Coins**

* **Relationship:** **Many-to-One**
* **Explanation:** Many assets (cryptocurrencies) can be associated with one coin type (e.g., Bitcoin, Ethereum), but each asset corresponds to a single coin type.

**5. Users and Withdrawals**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can make many withdrawals, but each withdrawal is linked to one user.

**6. Users and Watchlists**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can have many watchlists, but each watchlist belongs to a single user.

**7. Watchlists and Watchlist\_Coins**

* **Relationship:** **Many-to-Many**
* **Explanation:** A watchlist can contain many coins, and a coin can be part of many watchlists. This is a classic many-to-many relationship.

**8. Users and VerificationCodes**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can have multiple verification codes (e.g., OTPs for login or password recovery), but each verification code is linked to one user.

**9. Users and TradingHistories**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can have many trading history records, but each record is linked to one user.

**10. Users and PaymentOrders**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can make multiple payment orders, but each payment order is linked to one user.

**11. Users and PaymentDetails**

* **Relationship:** **One-to-One**
* **Explanation:** Each user has one set of payment details (bank account information), and each set of payment details belongs to only one user.

**12. Users and Orders**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can place many orders, but each order is placed by one user.

**13. Orders and OrderItems**

* **Relationship:** **One-to-Many**
* **Explanation:** One order can contain many items (e.g., multiple coins), but each order item belongs to one order.

**14. OrderItems and Coins**

* **Relationship:** **Many-to-One**
* **Explanation:** Many order items can be for the same coin, but each order item is associated with one specific coin.

**15. Users and Notifications**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can send or receive many notifications, but each notification is linked to a specific sender and receiver.

**16. MarketChartData**

* **Relationship:** **Independent**
* **Explanation:** MarketChartData stands on its own as it stores market data (prices) for coins over time. It doesn't directly link to any other entity in a one-to-one or one-to-many fashion.

**17. Users and ForgotPasswordTokens**

* **Relationship:** **One-to-Many**
* **Explanation:** A user can request multiple password recovery tokens, but each token is linked to one user.

**Visualizing Relationships:**

1. **One-to-One**:
   * **Users ↔ Wallets**
   * **Users ↔ PaymentDetails**
2. **One-to-Many**:
   * **Users ↔ WalletTransactions**
   * **Users ↔ Assets**
   * **Users ↔ Withdrawals**
   * **Users ↔ Watchlists**
   * **Users ↔ VerificationCodes**
   * **Users ↔ TradingHistories**
   * **Users ↔ PaymentOrders**
   * **Users ↔ Orders**
   * **Users ↔ Notifications**
   * **Users ↔ ForgotPasswordTokens**
   * **Orders ↔ OrderItems**
   * **Coins ↔ OrderItems**
3. **Many-to-Many**:
   * **Watchlists ↔ Watchlist\_Coins**

**Summary:**

* **One-to-One** relationships mean each instance of an entity is related to exactly one instance of another entity (e.g., Users ↔ PaymentDetails).
* **One-to-Many** relationships imply one entity can be associated with many instances of another entity (e.g., Users ↔ WalletTransactions).
* **Many-to-Many** relationships occur when multiple instances of one entity can relate to multiple instances of another entity (e.g., Watchlists ↔ Watchlist\_Coins).