Object Design

1. Classes and Attributes

1.1. User

Represents a system user, either a regular user or an admin.

- **Attributes**:
  - **userID**: Unique identifier (PK).
  - **name**: Name of the user.
  - **email**: Email address of the user.
  - **password**: Encrypted password.
  - **role**: Role of the user (User or Admin).
  - **createdAt**: Date and time when the user was created.

- **Methods**:
  - **authenticate()**: Validate user login credentials.
  - **resetPassword()**: Handle password reset.
  - **updateProfile()**: Update user details.

1.2. Portfolio

Represents a user's stock portfolio.

- **Attributes**:
  - **portfolioID:** Unique identifier (PK).
  - **userID**: Reference to the User who owns the portfolio (FK).
  - **stocks:** List of Stock objects in the portfolio.
  - **createdAt**: Date and time of portfolio creation.

- **Methods**:
  - **addStock(stock: Stock, quantity: int)**: Add a stock to the portfolio.
  - **removeStock(stockID: int):** Remove a stock from the portfolio.
  - **updateStockQuantity(stockID: int, quantity: int)**: Update stock quantity.
  - **calculatePerformance():** Compute portfolio performance metrics.

1.3. Stock

Represents an individual stock in the market or portfolio.

- **Attributes**:
  - **stockID**: Unique identifier (PK).

    - **name**: Stock name (e.g., "Apple Inc.").
    - **symbol**: Stock ticker symbol (e.g., "AAPL").
    - **currentPrice**: Current market price of the stock.
    - **lastUpdated**: Timestamp of the last price update.

- **Methods**:
    - **getMarketData()**: Fetch current stock data from an external API.
    - **updatePrice(newPrice: float):** Update the stock price.

1.4. Report

Represents a performance report for a user's portfolio.

- **Attributes**:
    - **reportID**: Unique identifier (PK).
    - **portfolioID**: Reference to the associated Portfolio (FK).
    - **reportData**: File path or generated report data (e.g., PDF/Excel).
    - **generatedAt**: Timestamp of report generation.

- **Methods**:
    - **generateReport()**: Create a report for the associated portfolio.

1.5. MachineLearningModel

Represents the machine learning model used for stock prediction.

- **Attributes**:
    - **modelID**: Unique identifier (PK).
    - **version**: Version of the ML model.
    - **accuracy**: Current accuracy of the model.
    - **lastTrained**: Timestamp of the last model training.

- **Methods**:
    - **predict(stockData: List[Stock])**: Generate stock performance predictions.
    - **retrainModel(trainingData: List[Stock]):** Retrain the model using new data.

1.6. Admin

Specialized user with privileges to manage users, stocks, and the ML model. Inherits from the User class.

- **Attributes**:
    - Inherits all attributes of the User class.

- **Methods**:
    - **addUser(user: User):** Add a new user.
    - **addStock(stock: Stock):** Add a new stock to the database.
    - **retrainModel():** Trigger retraining of the ML model.

2. Relationships Between Classes

2.1. **Relationships**:

- **User ↔ Portfolio:**
    - A user can have multiple portfolios (1-to-many).
    - Each portfolio belongs to one user.

- **Portfolio ↔ Stock:**
    - A portfolio can include multiple stocks (many-to-many relationship).
    - Stocks in a portfolio maintain quantities.

- **Portfolio ↔ Report:**
    - A portfolio can generate multiple reports (1-to-many).

- **Admin ↔ User:**
    - Admins manage users, but this is a logical association, not a direct object relationship.

- **Admin ↔ Stock:**
    - Admins manage stock data by adding stocks.

- **MachineLearningModel ↔ Stock:**
    - The ML model uses historical stock data for training and predictions.

```
Class User {
        +userID: int
        +name: string
        +email: string
        +password: string
        +role: string
        +createdAt: datetime
        +authenticate()
        +resetPassword()
        +updateProfile()
}

Class Admin extends User {
```

```
        +addUser()
        +addStock()
        +retraiModel()
}

Class Portfolio {
        +portfolioID: int
        +userID: int
        +stocks: List<Stock>
        +createdAt: datetime
        +addStock(stock, quantity)
   +removeStock(stockID)
   +updateStockQuantity(stockID, quantity)
   +calculatePerformance()
}

Class Stock {
        +stockID: int
        +name: string
        +symbol: string
        +currentPrice: float
        +lastUpdated: datetime
        +getMarketData()
        +updatePrice(newPrice)
}

Class Report {
        +reportID: int
        +portfolioID: int
        +reportData: string
        +generatedAt: datetime
        +generateReport()
}

Class MachineLearningModel {
        +modelID: int
        +version: string
        +accuracy: float
        +lastTrained: datetime
   +predict(stockData)
   +retrainModel(trainingData)
}
```