**Software Requirements Specification (SRS)**

**Portfolio Management System (PMS)**

**1. Introduction**

**1.1 Purpose**

The **Portfolio Management System (PMS)** is a **JDBC-based console application** designed to help investors manage their financial portfolios. It allows users to create and manage multiple portfolios, add investments (stocks/assets) to these portfolios, track their holdings, and perform CRUD operations on investors, portfolios, and stocks.

**1.2 Scope**

This application provides an easy-to-use interface for investors to organize and monitor their investments effectively. The system will:

* Allow **multiple investors** to register and manage their own portfolios.
* Support **multiple portfolios per investor** for different investment strategies.
* Enable investors to **add, update, and remove investments** (stocks/assets) in a portfolio.
* Allow real-time tracking of stock prices and portfolio valuation (optional future enhancement).
* Provide a **secure and structured database** for efficient data management.

**1.3 Intended Users**

* **Individual Investors** who want to track and manage their portfolios.
* **Financial Analysts** who need to organize and analyze multiple portfolios.
* **Investment Enthusiasts** looking for a simple way to manage their stock investments.

**1.4 Assumptions and Constraints**

* The application **does not have a front-end** and is purely a **console-based system**.
* The system uses **MySQL as the database** and JDBC for connectivity.
* The system **does not support real-time stock price updates** but allows manual updates of stock prices.
* The database structure **must enforce referential integrity** through foreign keys.
* Investors **cannot share portfolios**; each portfolio belongs to a single investor.

**2. System Overview**

**2.1 Key Features**

The application consists of four main modules:

**1. Investor Management**

* **Create an Investor**: Register a new investor in the system.
* **View Investor Details**: Retrieve investor details by ID.
* **Update Investor Information**: Modify name, email, phone number, and date of birth.
* **Delete Investor**: Remove an investor (along with their portfolios and investments).

**2. Portfolio Management**

* **Create Portfolio**: Allows an investor to create multiple portfolios with unique names.
* **View Portfolios**: Retrieve all portfolios associated with an investor.
* **Update Portfolio Name**: Change the name of an existing portfolio.
* **Delete Portfolio**: Remove a portfolio and its associated investments.

**3. Investment Management**

* **Add Investment**: Investors can buy stocks and allocate them to a portfolio.
* **View Investments**: Retrieve all investments within a portfolio.
* **Update Investment**: Modify the quantity and purchase price of a stock in a portfolio.
* **Delete Investment**: Remove an investment from a portfolio.

**4. Stock Management**

* **Add Stock**: Register new stocks/assets in the system.
* **View Stock Details**: Retrieve details of a stock by its ID.
* **Update Stock Price**: Modify the stock price (useful for market changes).
* **Delete Stock**: Remove a stock (only if it is not linked to any portfolio).

**3. Database Design**

The system follows a relational database structure with the following tables:

**3.1 Table Definitions**

| **Table Name** | **Description** |
| --- | --- |
|  |  |
| **investors** | Stores investor details like name, email, phone, and DOB. |
| **portfolio** | Stores portfolios linked to an investor. |
| **stocks** | Stores stock details like name, symbol, and price. |
| **portfolio\_investments** | Tracks which stocks belong to which portfolios. |

**3.2 Entity Relationships**

* **One investor can have multiple portfolios** (1:N relationship).
* **One portfolio can have multiple investments** (1:N relationship).
* **One stock can belong to multiple portfolios** (N:1 relationship).

**4. Functional Requirements**

**4.1 Investor Management**

| **ID** | **Requirement** |
| --- | --- |
| FR-1 | The system shall allow users to create an investor. |
| FR-2 | The system shall allow users to update investor details. |
| FR-3 | The system shall allow users to delete an investor, along with their portfolios and investments. |

**4.2 Portfolio Management**

| **ID** | **Requirement** |
| --- | --- |
| FR-4 | The system shall allow users to create multiple portfolios. |
| FR-5 | The system shall allow users to retrieve all portfolios by investor ID. |
| FR-6 | The system shall allow users to update a portfolio name. |
| FR-7 | The system shall allow users to delete a portfolio. |

**4.3 Investment Management**

| **ID** | **Requirement** |
| --- | --- |
| FR-8 | The system shall allow users to add stocks to a portfolio. |
| FR-9 | The system shall allow users to retrieve all investments in a portfolio. |
| FR-10 | The system shall allow users to update investment details. |
| FR-11 | The system shall allow users to delete an investment. |

**4.4 Stock Management**

| **ID** | **Requirement** |
| --- | --- |
| FR-12 | The system shall allow users to create new stocks. |
| FR-13 | The system shall allow users to retrieve stock details by ID. |
| FR-14 | The system shall allow users to update stock prices. |
| FR-15 | The system shall allow users to delete stocks (if they are not linked to portfolios). |

**5. System Flow**

**Investor Workflow**

1. **Register as an investor** → Enter name, email, phone, and date of birth.
2. **Create portfolios** → Each investor can create multiple portfolios.
3. **Add investments** → Assign stocks to portfolios and specify quantity & purchase price.
4. **View portfolios and investments** → List all holdings for tracking.
5. **Modify portfolios or investments** → Update names, quantities, or prices.
6. **Delete portfolios or investments** → Remove unwanted holdings.
7. **Delete investor account** → Remove investor data along with portfolios and investments.

**6. Non-Functional Requirements**

| **Category** | **Requirement** |
| --- | --- |
| **Performance** | The system should handle multiple investors and portfolios efficiently. |
| **Scalability** | The system should support expansion to new asset types (bonds, mutual funds, etc.). |
| **Security** | Data should be protected from SQL injection (using Prepared Statements). |
| **Reliability** | The system should prevent invalid transactions (e.g., negative stock quantity). |
| **Usability** | The system should provide clear console messages for user guidance. |

**7. Exception Handling**

| **Scenario** | **Handling Strategy** |
| --- | --- |
| Investor not found | Display an appropriate error message. |
| Portfolio does not exist | Prevent operations on non-existent portfolios. |
| Invalid stock ID | Reject investments in unregistered stocks. |
| Deleting a non-empty portfolio | Prevent deletion unless investments are removed first. |
| Database connection failure | Provide an error message and retry. |

**8. Conclusion**

The **Portfolio Management System (PMS)** is designed to help investors manage their financial holdings through **a simple JDBC-based application**. It provides an efficient way to track portfolios and investments while ensuring data integrity through structured relationships.

**Todo**:  
 Future enhancements may include **real-time stock price updates, reports, and a web-based UI.**

­­­