import java.util.\*;

import java.util.concurrent.ConcurrentHashMap;

class Stock {

String stockName;

double price;

List<Double> priceHistory = new ArrayList<>();

public Stock(String stockName, double price) {

this.stockName = stockName;

this.price = price;

priceHistory.add(price);

}

public void updatePrice(double newPrice) {

this.price = newPrice;

priceHistory.add(newPrice);

}

public double calculateSMA(int period) {

if (priceHistory.size() < period) return -1;

double sum = 0;

for (int i = priceHistory.size() - period; i < priceHistory.size(); i++) {

sum += priceHistory.get(i);

}

return sum / period;

}

public double calculateEMA(int period) {

if (priceHistory.size() < period) return -1;

double k = 2.0 / (period + 1);

double ema = priceHistory.get(priceHistory.size() - period);

for (int i = priceHistory.size() - period + 1; i < priceHistory.size(); i++) {

ema = priceHistory.get(i) \* k + ema \* (1 - k);

}

return ema;

}

}

class User {

String username;

String password;

double balance;

Map<String, Integer> portfolio = new HashMap<>();

List<String> transactionHistory = new ArrayList<>();

public User(String username, String password, double balance) {

this.username = username;

this.password = password;

this.balance = balance;

}

public void addToPortfolio(String stockSymbol, int quantity) {

portfolio.put(stockSymbol, portfolio.getOrDefault(stockSymbol, 0) + quantity);

}

public void removeFromPortfolio(String stockSymbol, int quantity) {

portfolio.put(stockSymbol, portfolio.get(stockSymbol) - quantity);

if (portfolio.get(stockSymbol) <= 0) {

portfolio.remove(stockSymbol);

}

}

}

class StockTradingSystem {

static Scanner scanner = new Scanner(System.in);

static Map<String, Stock> market = new ConcurrentHashMap<>();

static Map<String, User> users = new HashMap<>();

static List<String> newsFeed = new ArrayList<>();

public static void main(String[] args) {

initializeMarket();

startMarketSimulation();

while (true) {

System.out.println("1. Register\n2. Login\n3. Exit");

int choice = getIntInput();

if (choice == 1) {

registerUser();

} else if (choice == 2) {

User user = loginUser();

if (user != null) {

displayUserMenu(user);

}

} else if (choice == 3) {

System.out.println("Exiting...");

break;

}

}

scanner.close();

}

static void initializeMarket() {

market.put("AAPL", new Stock("AAPL", 150));

market.put("TSLA", new Stock("TSLA", 800));

market.put("GOOG", new Stock("GOOG", 2800));

market.put("MSFT", new Stock("MSFT", 300));

newsFeed.add("Apple announces new iPhone.");

newsFeed.add("Tesla achieves record sales.");

newsFeed.add("Google acquires AI startup.");

newsFeed.add("Microsoft launches new Surface product.");

}

static void registerUser() {

System.out.print("Enter username: ");

String username = scanner.next();

System.out.print("Enter password: ");

String password = scanner.next();

users.put(username, new User(username, password, 10000));

System.out.println("User registered successfully!");

}

static User loginUser() {

System.out.print("Enter username: ");

String username = scanner.next();

System.out.print("Enter password: ");

String password = scanner.next();

User user = users.get(username);

if (user != null && user.password.equals(password)) {

System.out.println("Login successful!");

return user;

} else {

System.out.println("Invalid username or password.");

return null;

}

}

static void displayUserMenu(User user) {

while (true) {

System.out.println("\n1. View Portfolio\n2. View Market\n3. Buy Stock\n4. Sell Stock\n5. Add Funds");

System.out.println("6. View Market News\n7. View Technical Indicators\n8. Logout");

int choice = getIntInput();

switch (choice) {

case 1: viewPortfolio(user); break;

case 2: viewMarket(); break;

case 3: buyStock(user); break;

case 4: sellStock(user); break;

case 5: addFunds(user); break;

case 6: displayMarketNews(); break;

case 7: viewTechnicalIndicators(user); break;

case 8: System.out.println("Logging out..."); return;

default: System.out.println("Invalid choice. Try again.");

}

}

}

static void viewPortfolio(User user) {

System.out.println("\nPortfolio:");

double totalValue = 0;

for (Map.Entry<String, Integer> entry : user.portfolio.entrySet()) {

Stock stock = market.get(entry.getKey());

if (stock != null) {

System.out.println(entry.getKey() + ": " + entry.getValue() + " shares at $" + stock.price);

totalValue += stock.price \* entry.getValue();

}

}

System.out.println("Total Portfolio Value: $" + totalValue);

}

static void viewMarket() {

System.out.println("\nMarket Prices:");

for (Stock stock : market.values()) {

System.out.println(stock.stockName + ": $" + stock.price);

}

}

static void buyStock(User user) {

System.out.print("\nEnter stock symbol to buy: ");

String stockSymbol = scanner.next();

System.out.print("Enter quantity: ");

int quantity = getIntInput();

Stock stock = market.get(stockSymbol);

if (stock != null) {

double totalPrice = stock.price \* quantity;

if (user.balance >= totalPrice) {

user.balance -= totalPrice;

user.addToPortfolio(stockSymbol, quantity);

user.transactionHistory.add("Bought " + quantity + " shares of " + stockSymbol);

System.out.println("Bought " + quantity + " shares of " + stockSymbol + " for $" + totalPrice);

} else {

System.out.println("Insufficient balance.");

}

} else {

System.out.println("Stock not found.");

}

}

static void sellStock(User user) {

System.out.print("\nEnter stock symbol to sell: ");

String stockSymbol = scanner.next();

System.out.print("Enter quantity: ");

int quantity = getIntInput();

if (user.portfolio.getOrDefault(stockSymbol, 0) >= quantity) {

Stock stock = market.get(stockSymbol);

double totalPrice = stock.price \* quantity;

user.balance += totalPrice;

user.removeFromPortfolio(stockSymbol, quantity);

user.transactionHistory.add("Sold " + quantity + " shares of " + stockSymbol);

System.out.println("Sold " + quantity + " shares of " + stockSymbol + " for $" + totalPrice);

} else {

System.out.println("Insufficient shares in portfolio.");

}

}

static void addFunds(User user) {

System.out.print("\nEnter amount to add: ");

double amount = scanner.nextDouble();

user.balance += amount;

System.out.println("New balance: $" + user.balance);

}

static void displayMarketNews() {

System.out.println("\nMarket News:");

for (String news : newsFeed) {

System.out.println(news);

}

}

static int getIntInput() {

while (!scanner.hasNextInt()) scanner.next();

return scanner.nextInt();

}

static void startMarketSimulation() {}

static void viewTechnicalIndicators(User user) {}

}