Comprendre une plateforme de gestion des actions en bourse

Une plateforme de gestion des actions en bourse est essentiellement un site web qui permet aux utilisateurs de suivre, analyser et parfois simuler des transactions

boursières. Voici les fonctionnalités typiques qu'on y trouve :

1. Un tableau de bord présentant une vue d'ensemble du portefeuille de l'utilisateur
2. Des graphiques montrant l'évolution des cours des actions
3. Des outils de recherche pour trouver des informations sur différentes entreprises
4. Des mécanismes pour "acheter" et "vendre" virtuellement des actions
5. Des analyses et statistiques sur les performances du portefeuille

À quoi ressemble ce genre de site

Visuellement, ces plateformes comportent généralement :

- Des tableaux de données avec les cours des actions
- Des graphiques interactifs montrant l'évolution des prix
- Des formulaires pour effectuer des transactions
- Des pages de détails sur chaque action ou entreprise
- Un système d'authentification pour que chaque utilisateur gère son propre portefeuille

Étapes pour réaliser ce projet

Je vais vous proposer une démarche complète pour développer cette plateforme :

Étape 1 : Analyse et planification

Commencez par définir précisément ce que votre application doit faire :

- Quelles fonctionnalités sont requises dans votre cahier des charges ?
- S'agit-il d'une simulation ou connecterez-vous l'application à des données réelles ?
- Quels sont les utilisateurs cibles et leurs besoins ?

Créez ensuite une maquette (wireframe) de votre application pour visualiser l'interface et l'expérience utilisateur.

Étape 2 : Mise en place de l'environnement de développement

Choisissez les technologies que vous allez utiliser. Voici une pile technologique possible :

- **Frontend** : HTML, CSS, JavaScript avec un framework comme React, Vue.js ou Angular
- **Backend** : Node.js, Python (Django/Flask), PHP ou autre selon vos compétences
- **Base de données** : MySQL, PostgreSQL ou MongoDB
- **API de données boursières** : Yahoo Finance, Alpha Vantage (gratuit avec limitations), ou une API simulée

Étape 3 : Développement du backend

Créez la structure de votre base de données pour stocker :

- Les informations utilisateurs
- Les données des actions (symbole, nom, secteur, etc.)
- Les portefeuilles des utilisateurs
- L'historique des transactions

Développez les API nécessaires :

- Authentification et gestion des utilisateurs
- Récupération des données boursières (réelles ou simulées)
- Gestion des transactions (achat/vente)
- Calcul des performances et statistiques

Étape 4 : Développement du frontend

Créez les interfaces utilisateur principales :

1. **Page d'accueil et inscription/connexion**
2. **Tableau de bord principal** montrant le portefeuille et sa performance
3. **Page de recherche d'actions** avec filtres et informations de base

4. **Page détaillée d'une action** avec graphiques et données fondamentales
5. **Interface de transaction** pour acheter/vendre des actions
6. **Historique des transactions** et analyses de performance

Étape 5 : Fonctionnalités avancées

Selon les exigences de votre projet, vous pourriez ajouter :

- Des alertes de prix
- Des indicateurs techniques (moyennes mobiles, RSI, etc.)
- Des comparaisons entre différentes actions
- Des simulations de stratégies d'investissement
- Des fonctionnalités sociales (partage de portefeuilles, classements)

Étape 6 : Tests et déploiement

Testez votre application pour vous assurer que :

- Toutes les fonctionnalités marchent correctement
- L'interface est responsive (s'adapte aux mobiles)
- Les calculs financiers sont exacts
- La sécurité est adéquate

Exemple de code pour démarrer

Voici un exemple simple de structure frontend pour votre tableau de bord :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>StockManager - Tableau de bord</title>
    <link rel="stylesheet" href="styles.css">
    <!-- Inclusion de Chart.js pour les graphiques -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <header>
        <nav>
```

```
            <div class="logo">StockManager</div>
            <ul>
                <li><a href="#" class="active">Tableau de bord</a></li>
                <li><a href="#">Recherche</a></li>
                <li><a href="#">Transactions</a></li>
                <li><a href="#">Analyses</a></li>
                <li><a href="#" class="user-profile">Mon Profil</a></li>
            </ul>
        </nav>
    </header>

    <main>
        <section class="portfolio-summary">
            <h2>Mon Portefeuille</h2>
            <div class="portfolio-stats">
                <div class="stat-card">
                    <h3>Valeur Totale</h3>
                    <p class="value">10 245,67 €</p>
                    <p class="change positive">+2,4% aujourd'hui</p>
                </div>
                <div class="stat-card">
                    <h3>Rendement Total</h3>
                    <p class="value">+824,30 €</p>
                    <p class="change positive">+8,7% depuis le début</p>
                </div>
                <div class="stat-card">
                    <h3>Liquidités</h3>
                    <p class="value">1 234,56 €</p>
                </div>
            </div>

            <div class="portfolio-chart">
                <canvas id="portfolioChart"></canvas>
            </div>
        </section>

        <section class="holdings">
            <h2>Mes Actions</h2>
            <table class="holdings-table">
                <thead>
                    <tr>
                        <th>Symbole</th>
                        <th>Nom</th>
                        <th>Quantité</th>
                        <th>Prix Moyen</th>
                        <th>Prix Actuel</th>
                        <th>Variation (jour)</th>
                        <th>Valeur</th>
                        <th>Gain/Perte</th>
                        <th>Actions</th>
                    </tr>
                </thead>
                <tbody>
                    <tr>
                        <td>AAPL</td>
                        <td>Apple Inc.</td>
                        <td>10</td>
```

```
                <td>150,25 €</td>
                <td>165,30 €</td>
                <td class="positive">+1,2%</td>
                <td>1 653,00 €</td>
                <td class="positive">+150,50 € (+10,0%)</td>
                <td>
                    <button class="btn-buy">Acheter</button>
                    <button class="btn-sell">Vendre</button>
                </td>
            </tr>
            <tr>
                <td>MSFT</td>
                <td>Microsoft Corp.</td>
                <td>5</td>
                <td>280,40 €</td>
                <td>290,75 €</td>
                <td class="positive">+0,8%</td>
                <td>1 453,75 €</td>
                <td class="positive">+51,75 € (+3,7%)</td>
                <td>
                    <button class="btn-buy">Acheter</button>
                    <button class="btn-sell">Vendre</button>
                </td>
            </tr>
            <tr>
                <td>TSLA</td>
                <td>Tesla Inc.</td>
                <td>3</td>
                <td>900,10 €</td>
                <td>820,50 €</td>
                <td class="negative">-2,1%</td>
                <td>2 461,50 €</td>
                <td class="negative">-238,80 € (-8,8%)</td>
                <td>
                    <button class="btn-buy">Acheter</button>
                    <button class="btn-sell">Vendre</button>
                </td>
            </tr>
        </tbody>
    </table>
</section>

<section class="market-overview">
    <h2>Aperçu du Marché</h2>
    <div class="index-cards">
        <div class="index-card">
            <h3>CAC 40</h3>
            <p class="value">7 245,34</p>
            <p class="change positive">+0,5%</p>
        </div>
        <div class="index-card">
            <h3>S&P 500</h3>
            <p class="value">4 532,76</p>
            <p class="change positive">+0,3%</p>
        </div>
        <div class="index-card">
            <h3>NASDAQ</h3>
```

```
                    <p class="value">14 893,45</p>
                    <p class="change negative">-0,2%</p>
                </div>
            </div>
        </section>
    </main>

    <footer>
        <p>© 2025 StockManager - Projet Académique</p>
        <p>Les données sont simulées et ne représentent pas des valeurs réelles du
marché</p>
    </footer>

    <script src="dashboard.js"></script>
</body>
</html>
```

Voici le CSS correspondant pour donner un style moderne à votre interface :

```
/* Reset et styles de base */
* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
}

body {
    background-color: #f5f7fa;
    color: #333;
    line-height: 1.6;
}

/* En-tête et navigation */
header {
    background-color: #fff;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);
    position: sticky;
    top: 0;
    z-index: 100;
}

nav {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 1rem 2rem;
    max-width: 1400px;
    margin: 0 auto;
}

.logo {
    font-size: 1.5rem;
```

```
    font-weight: bold;
    color: #2c3e50;
}

nav ul {
    display: flex;
    list-style: none;
}

nav ul li {
    margin-left: 2rem;
}

nav a {
    text-decoration: none;
    color: #7f8c8d;
    font-weight: 500;
    transition: color 0.3s;
}

nav a:hover, nav a.active {
    color: #3498db;
}

.user-profile {
    background-color: #3498db;
    color: white !important;
    padding: 0.5rem 1rem;
    border-radius: 4px;
}

/* Conteneur principal */
main {
    max-width: 1400px;
    margin: 2rem auto;
    padding: 0 2rem;
}

section {
    background-color: #fff;
    border-radius: 8px;
    box-shadow: 0 2px 10px rgba(0, 0, 0, 0.05);
    padding: 1.5rem;
    margin-bottom: 2rem;
}

h2 {
    color: #2c3e50;
    margin-bottom: 1.5rem;
    font-weight: 600;
}

/* Cartes statistiques */
.portfolio-stats, .index-cards {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
    gap: 1.5rem;
```

```
    margin-bottom: 2rem;
}

.stat-card, .index-card {
    background-color: #f8f9fa;
    border-radius: 8px;
    padding: 1.5rem;
    text-align: center;
    transition: transform 0.3s, box-shadow 0.3s;
}

.stat-card:hover, .index-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 5px 15px rgba(0, 0, 0, 0.1);
}

.stat-card h3, .index-card h3 {
    font-size: 1rem;
    color: #7f8c8d;
    margin-bottom: 0.5rem;
}

.value {
    font-size: 1.8rem;
    font-weight: bold;
    color: #2c3e50;
    margin-bottom: 0.5rem;
}

.change {
    font-size: 0.9rem;
    font-weight: 500;
}

.positive {
    color: #27ae60;
}

.negative {
    color: #e74c3c;
}

/* Graphique du portefeuille */
.portfolio-chart {
    width: 100%;
    height: 300px;
    margin-top: 2rem;
}

/* Tableau des actions */
.holdings-table {
    width: 100%;
    border-collapse: collapse;
    margin-top: 1rem;
}

.holdings-table th,
```

```css
.holdings-table td {
    padding: 1rem;
    text-align: left;
    border-bottom: 1px solid #ecf0f1;
}

.holdings-table th {
    background-color: #f8f9fa;
    color: #7f8c8d;
    font-weight: 600;
}

.holdings-table tr:hover {
    background-color: #f8f9fa;
}

/* Boutons */
button {
    padding: 0.5rem 1rem;
    border: none;
    border-radius: 4px;
    font-weight: 500;
    cursor: pointer;
    transition: background-color 0.3s;
}

.btn-buy {
    background-color: #3498db;
    color: white;
    margin-right: 0.5rem;
}

.btn-sell {
    background-color: #e74c3c;
    color: white;
}

.btn-buy:hover {
    background-color: #2980b9;
}

.btn-sell:hover {
    background-color: #c0392b;
}

/* Pied de page */
footer {
    text-align: center;
    padding: 2rem;
    background-color: #2c3e50;
    color: #ecf0f1;
    font-size: 0.9rem;
}

footer p {
    margin: 0.5rem 0;
}
```

```
/* Responsive */
@media (max-width: 768px) {
    nav {
        flex-direction: column;
    }

    nav ul {
        margin-top: 1rem;
        flex-wrap: wrap;
    }

    nav ul li {
        margin: 0.5rem;
    }

    .holdings-table {
        overflow-x: auto;
        display: block;
    }
}
```

Et pour finir, voici un exemple de code JavaScript pour initialiser le graphique :

```
// Données pour le graphique d'évolution du portefeuille
const portfolioData = {
    labels: ['Jan', 'Fév', 'Mar', 'Avr', 'Mai', 'Juin', 'Juil'],
    datasets: [{
        label: 'Valeur du portefeuille (€)',
        data: [8500, 8700, 9200, 9100, 9600, 9800, 10245.67],
        backgroundColor: 'rgba(52, 152, 219, 0.1)',
        borderColor: 'rgba(52, 152, 219, 1)',
        borderWidth: 2,
        pointBackgroundColor: 'rgba(52, 152, 219, 1)',
        pointRadius: 4,
        tension: 0.2 // Pour avoir des courbes plus douces
    }]
};

// Configuration du graphique
const portfolioChartConfig = {
    type: 'line',
    data: portfolioData,
    options: {
        responsive: true,
        maintainAspectRatio: false,
        plugins: {
            legend: {
                display: true,
                position: 'top'
            },
            tooltip: {
                mode: 'index',
```

```
                intersect: false,
                callbacks: {
                    label: function(context) {
                        let label = context.dataset.label || '';
                        if (label) {
                            label += ': ';
                        }
                        if (context.parsed.y !== null) {
                            label += new Intl.NumberFormat('fr-FR', {
                                style: 'currency',
                                currency: 'EUR'
                            }).format(context.parsed.y);
                        }
                        return label;
                    }
                }
            }
        },
        scales: {
            x: {
                grid: {
                    display: false
                }
            },
            y: {
                beginAtZero: false,
                ticks: {
                    callback: function(value) {
                        return new Intl.NumberFormat('fr-FR', {
                            style: 'currency',
                            currency: 'EUR',
                            maximumFractionDigits: 0
                        }).format(value);
                    }
                }
            }
        }
    }
};

// Initialiser le graphique lorsque le DOM est chargé
document.addEventListener('DOMContentLoaded', function() {
    const ctx = document.getElementById('portfolioChart').getContext('2d');
    new Chart(ctx, portfolioChartConfig);

    // Ajouter des écouteurs d'événements pour les boutons
    document.querySelectorAll('.btn-buy').forEach(button => {
        button.addEventListener('click', function(e) {
            const row = e.target.closest('tr');
            const symbol = row.cells[0].textContent;
            const name = row.cells[1].textContent;
            openTransactionModal('buy', symbol, name);
        });
    });

    document.querySelectorAll('.btn-sell').forEach(button => {
        button.addEventListener('click', function(e) {
```

```
            const row = e.target.closest('tr');
            const symbol = row.cells[0].textContent;
            const name = row.cells[1].textContent;
            openTransactionModal('sell', symbol, name);
        });
    });
});

// Fonction pour ouvrir une modale de transaction (à implémenter)
function openTransactionModal(type, symbol, name) {
    console.log(`Ouvrir modale pour ${type === 'buy' ? 'acheter' : 'vendre'}
${name} (${symbol})`);
    // Ici vous ajouteriez le code pour ouvrir une modale avec un formulaire de
transaction
    alert(`${type === 'buy' ? 'Achat' : 'Vente'} de ${name} (${symbol})`);
}
```

Structure de la base de données

Pour votre application, voici comment vous pourriez structurer votre base de données :

Table `users`

- id (clé primaire)
- username
- email
- password (haché)
- date_created
- balance (solde disponible)

Table `stocks`

- id (clé primaire)
- symbol (ex: AAPL)
- name (ex: Apple Inc.)
- sector (ex: Technology)
- current_price
- last_updated

Table `user_portfolios`

- id (clé primaire)
- user_id (clé étrangère)
- stock_id (clé étrangère)
- quantity
- average_buy_price

Table `transactions`

- id (clé primaire)
- user_id (clé étrangère)
- stock_id (clé étrangère)
- type (achat/vente)
- quantity
- price
- timestamp

Comment gérer les données boursières

Vous avez plusieurs options pour gérer les données boursières dans votre application :

1. **Données simulées** : Le plus simple pour un projet académique. Créez un jeu de données statique ou avec des variations aléatoires.

2. **API gratuites avec limitations** :

    - Yahoo Finance API
    - Alpha Vantage (limite d'appels par jour)
    - FinnHub (limite d'appels par minute)

3. **Fichiers CSV historiques** : Si vous n'avez pas besoin de données en temps réel, vous pouvez télécharger des fichiers CSV d'historiques boursiers et les importer dans votre base de données.

Planification du développement

Voici un planning réaliste pour développer ce projet :

Planning sur 5 semaine

Semaine 1 : Analyse et conception

- Définir les fonctionnalités précises
- Créer des wireframes et maquettes
- Planifier l'architecture technique

Semaine 2 : Mise en place de l'environnement et base de données

- Configurer l'environnement de développement
- Créer la structure de la base de données
- Implémenter le système d'authentification

Semaine 3 : Backend - API et logique métier

- Développer les API pour récupérer les données boursières
- Implémenter la logique d'achat/vente
- Créer les calculs pour le suivi de performance

Semaine 4 : Frontend - Interface utilisateur

- Développer le tableau de bord principal
- Créer l'interface de recherche et de détail des actions
- Implémenter l'interface de transaction

Semaine 5 : Tests et finalisation

- Tester toutes les fonctionnalités
- Corriger les bugs
- Polir l'interface utilisateur
- Préparer la documentation

Ce projet vous permettra d'acquérir des compétences précieuses en développement web, notamment dans la gestion d'interfaces complexes, la visualisation de données et l'architecture d'applications. N'hésitez pas à me demander des précisions sur n'importe quel aspect du projet ! Avec ces contraintes de temps et de compétences, je vais

adapter le projet pour qu'il soit réalisable en 2-3 semaines avec une équipe de trois personnes dont deux ont des connaissances intermédiaires en développement web.

Projet simplifié : Plateforme de gestion d'actions en bourse

Répartition du travail

Puisque vous êtes trois, je vous suggère cette répartition :

- Personne 1 (avec connaissances) : Frontend et intégration
- Personne 2 (avec connaissances) : Backend et base de données
- Personne 3 : Maquettes, tests, documentation et support

Fonctionnalités essentielles pour un projet en 2-3 semaines

Pour rendre le projet réalisable dans ce délai, concentrez-vous sur ces fonctionnalités clés :

1. **Authentification basique** (inscription/connexion)
2. **Tableau de bord** montrant le portefeuille de l'utilisateur
3. **Recherche simple** d'actions par nom ou symbole
4. **Achat/vente simulés** d'actions
5. **Historique des transactions**
6. **Visualisation simple** de l'évolution du portefeuille

Technologie recommandée pour un développement rapide

Pour accélérer le développement avec des connaissances intermédiaires, je recommande :

- **Frontend** : HTML, CSS, JavaScript avec Bootstrap ou Tailwind pour le design
- **Backend** : PHP avec un framework léger comme Slim, ou Node.js avec Express
- **Base de données** : MySQL ou SQLite (plus simple à configurer)
- **Données boursières** : Utiliser des données simulées ou statiques (pas d'API en temps réel)

Planning sur 2 semaines

Semaine 1

Jours 1-2 : Planification et configuration

- Définir ensemble les fonctionnalités précises
- Créer des wireframes simples
- Configurer l'environnement de développement
- Créer la structure de la base de données

Jours 3-5 : Développement des fondations

- Frontend : Structure HTML des pages principales et styles CSS
- Backend : Système d'authentification et routes API de base
- Intégration : Connexion du frontend au backend

Semaine 2

Jours 6-8 : Fonctionnalités principales

- Frontend : Finalisation des interfaces et intégration des graphiques
- Backend : Logique d'achat/vente et calcul des performances
- Tests : Vérification des fonctionnalités

Jours 9-10 : Finalisation

- Correction des bugs
- Documentation
- Préparation de la présentation

Version simplifiée du code

Voici une version simplifiée du tableau de bord, plus facile à implémenter :

```
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>StockManager - Tableau de bord</title>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.css" rel="stylesheet">
    <!-- Chart.js -->
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
        <div class="container">
            <a class="navbar-brand" href="#">StockManager</a>
            <button class="navbar-toggler" type="button" data-bs-toggle="collapse"
data-bs-target="#navbarNav">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNav">
                <ul class="navbar-nav me-auto">
                    <li class="nav-item">
                        <a class="nav-link active" href="#">Tableau de bord</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="#">Recherche</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="#">Transactions</a>
                    </li>
                </ul>
                <span class="navbar-text">
                    Bienvenue, Utilisateur | <a href="#" class="text-
white">Déconnexion</a>
                </span>
            </div>
        </div>
    </nav>

    <div class="container mt-4">
        <!-- Résumé du portefeuille -->
        <div class="row mb-4">
            <div class="col-md-4">
                <div class="card">
                    <div class="card-body text-center">
                        <h5 class="card-title">Valeur Totale</h5>
                        <h2 class="card-text">10 245,67 €</h2>
                        <p class="text-success">+2,4% aujourd'hui</p>
```

```
                    </div>
                </div>
            </div>
            <div class="col-md-4">
                <div class="card">
                    <div class="card-body text-center">
                        <h5 class="card-title">Rendement Total</h5>
                        <h2 class="card-text">+824,30 €</h2>
                        <p class="text-success">+8,7% depuis le début</p>
                    </div>
                </div>
            </div>
            <div class="col-md-4">
                <div class="card">
                    <div class="card-body text-center">
                        <h5 class="card-title">Liquidités</h5>
                        <h2 class="card-text">1 234,56 €</h2>
                    </div>
                </div>
            </div>
        </div>

        <!-- Graphique du portefeuille -->
        <div class="card mb-4">
            <div class="card-header">
                Évolution du portefeuille
            </div>
            <div class="card-body">
                <canvas id="portfolioChart" height="250"></canvas>
            </div>
        </div>

        <!-- Tableau des actions -->
        <div class="card mb-4">
            <div class="card-header d-flex justify-content-between align-items-
center">
                <span>Mes Actions</span>
                <button class="btn btn-primary btn-sm" data-bs-toggle="modal" data-
bs-target="#searchModal">
                    Acheter des actions
                </button>
            </div>
            <div class="card-body">
                <div class="table-responsive">
                    <table class="table table-hover">
                        <thead>
                            <tr>
                                <th>Symbole</th>
                                <th>Entreprise</th>
                                <th>Quantité</th>
                                <th>Prix d'achat</th>
                                <th>Prix actuel</th>
                                <th>Valeur</th>
                                <th>Gain/Perte</th>
                                <th>Actions</th>
                            </tr>
                        </thead>
```

```
                        <tbody>
                            <tr>
                                <td>AAPL</td>
                                <td>Apple Inc.</td>
                                <td>10</td>
                                <td>150,25 €</td>
                                <td>165,30 €</td>
                                <td>1 653,00 €</td>
                                <td class="text-success">+150,50 € (+10,0%)</td>
                                <td>
                                    <button class="btn btn-success btn-
sm">Acheter</button>
                                    <button class="btn btn-danger btn-
sm">Vendre</button>
                                </td>
                            </tr>
                            <tr>
                                <td>MSFT</td>
                                <td>Microsoft Corp.</td>
                                <td>5</td>
                                <td>280,40 €</td>
                                <td>290,75 €</td>
                                <td>1 453,75 €</td>
                                <td class="text-success">+51,75 € (+3,7%)</td>
                                <td>
                                    <button class="btn btn-success btn-
sm">Acheter</button>
                                    <button class="btn btn-danger btn-
sm">Vendre</button>
                                </td>
                            </tr>
                            <tr>
                                <td>TSLA</td>
                                <td>Tesla Inc.</td>
                                <td>3</td>
                                <td>900,10 €</td>
                                <td>820,50 €</td>
                                <td>2 461,50 €</td>
                                <td class="text-danger">-238,80 € (-8,8%)</td>
                                <td>
                                    <button class="btn btn-success btn-
sm">Acheter</button>
                                    <button class="btn btn-danger btn-
sm">Vendre</button>
                                </td>
                            </tr>
                        </tbody>
                    </table>
                </div>
            </div>
        </div>

        <!-- Dernières transactions -->
        <div class="card">
            <div class="card-header">
                Dernières Transactions
            </div>
```

```
            <div class="card-body">
                <div class="table-responsive">
                    <table class="table table-sm">
                        <thead>
                            <tr>
                                <th>Date</th>
                                <th>Type</th>
                                <th>Symbole</th>
                                <th>Quantité</th>
                                <th>Prix</th>
                                <th>Total</th>
                            </tr>
                        </thead>
                        <tbody>
                            <tr>
                                <td>01/03/2025</td>
                                <td class="text-success">Achat</td>
                                <td>AAPL</td>
                                <td>2</td>
                                <td>165,30 €</td>
                                <td>330,60 €</td>
                            </tr>
                            <tr>
                                <td>28/02/2025</td>
                                <td class="text-danger">Vente</td>
                                <td>GOOG</td>
                                <td>1</td>
                                <td>2 730,20 €</td>
                                <td>2 730,20 €</td>
                            </tr>
                            <tr>
                                <td>25/02/2025</td>
                                <td class="text-success">Achat</td>
                                <td>MSFT</td>
                                <td>3</td>
                                <td>290,75 €</td>
                                <td>872,25 €</td>
                            </tr>
                        </tbody>
                    </table>
                </div>
            </div>
        </div>
    </div>

    <!-- Modal de recherche d'actions -->
    <div class="modal fade" id="searchModal" tabindex="-1">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <h5 class="modal-title">Rechercher une action</h5>
                    <button type="button" class="btn-close" data-bs-
dismiss="modal"></button>
                </div>
                <div class="modal-body">
                    <div class="mb-3">
                        <input type="text" class="form-control" placeholder="Nom ou
```

```
symbole de l'entreprise">
                    </div>
                    <div class="list-group">
                        <button class="list-group-item list-group-item-action">
                            <div class="d-flex justify-content-between">
                                <div>
                                    <strong>AAPL</strong> - Apple Inc.
                                </div>
                                <div>165,30 €</div>
                            </div>
                        </button>
                        <button class="list-group-item list-group-item-action">
                            <div class="d-flex justify-content-between">
                                <div>
                                    <strong>MSFT</strong> - Microsoft Corp.
                                </div>
                                <div>290,75 €</div>
                            </div>
                        </button>
                        <button class="list-group-item list-group-item-action">
                            <div class="d-flex justify-content-between">
                                <div>
                                    <strong>GOOG</strong> - Alphabet Inc.
                                </div>
                                <div>2 730,20 €</div>
                            </div>
                        </button>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <!-- Bootstrap JS -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.min.js"></script>

    <!-- Script pour le graphique -->
    <script>
        document.addEventListener('DOMContentLoaded', function() {
            const ctx = document.getElementById('portfolioChart').getContext('2d');
            new Chart(ctx, {
                type: 'line',
                data: {
                    labels: ['Jan', 'Fév', 'Mar', 'Avr', 'Mai', 'Juin', 'Juil'],
                    datasets: [{
                        label: 'Valeur du portefeuille (€)',
                        data: [8500, 8700, 9200, 9100, 9600, 9800, 10245.67],
                        backgroundColor: 'rgba(13, 110, 253, 0.1)',
                        borderColor: 'rgba(13, 110, 253, 1)',
                        borderWidth: 2,
                        tension: 0.1
                    }]
                },
                options: {
                    responsive: true,
                    plugins: {
```

```
                        legend: {
                            position: 'top',
                        }
                    },
                    scales: {
                        y: {
                            beginAtZero: false
                        }
                    }
                }
            });
        });
    </script>
</body>
</html>
```

Structure de base de données simplifiée

Pour simplifier le projet, limitez votre base de données à ces tables essentielles :

```
-- Table des utilisateurs
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL UNIQUE,
    email VARCHAR(100) NOT NULL UNIQUE,
    password VARCHAR(255) NOT NULL,
    balance DECIMAL(10, 2) DEFAULT 10000.00,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Table des actions
CREATE TABLE stocks (
    id INT AUTO_INCREMENT PRIMARY KEY,
    symbol VARCHAR(10) NOT NULL UNIQUE,
    name VARCHAR(100) NOT NULL,
    current_price DECIMAL(10, 2) NOT NULL,
    last_updated TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);

-- Table du portefeuille utilisateur
CREATE TABLE portfolio (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    stock_id INT NOT NULL,
    quantity INT NOT NULL,
    avg_purchase_price DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (user_id) REFERENCES users(id),
    FOREIGN KEY (stock_id) REFERENCES stocks(id),
    UNIQUE KEY user_stock (user_id, stock_id)
);
```

```sql
-- Table des transactions
CREATE TABLE transactions (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT NOT NULL,
    stock_id INT NOT NULL,
    type ENUM('buy', 'sell') NOT NULL,
    quantity INT NOT NULL,
    price DECIMAL(10, 2) NOT NULL,
    total_amount DECIMAL(10, 2) NOT NULL,
    transaction_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (user_id) REFERENCES users(id),
    FOREIGN KEY (stock_id) REFERENCES stocks(id)
);

-- Données de démonstration pour les actions
INSERT INTO stocks (symbol, name, current_price) VALUES
('AAPL', 'Apple Inc.', 165.30),
('MSFT', 'Microsoft Corporation', 290.75),
('GOOG', 'Alphabet Inc.', 2730.20),
('AMZN', 'Amazon.com Inc.', 3350.50),
('TSLA', 'Tesla Inc.', 820.50),
('META', 'Meta Platforms Inc.', 335.80),
('NFLX', 'Netflix Inc.', 610.25),
('NVDA', 'NVIDIA Corporation', 775.40),
('PEP', 'PepsiCo Inc.', 172.60),
('KO', 'The Coca-Cola Company', 60.25);
```

Fonctionnalité d'achat/vente

Voici un exemple simplifié de fonction PHP pour gérer les transactions :

```php
<?php
// Fonction pour acheter des actions
function buyStock($userId, $stockId, $quantity, $pdo) {
    try {
        // Début de la transaction pour garantir l'intégrité des données
        $pdo->beginTransaction();

        // 1. Obtenir le prix actuel de l'action et les informations utilisateur
        $stmt = $pdo->prepare("SELECT current_price FROM stocks WHERE id = ?");
        $stmt->execute([$stockId]);
        $stock = $stmt->fetch();

        $stmt = $pdo->prepare("SELECT balance FROM users WHERE id = ?");
        $stmt->execute([$userId]);
        $user = $stmt->fetch();

        // 2. Calculer le coût total
        $totalCost = $stock['current_price'] * $quantity;
```

```
        // 3. Vérifier si l'utilisateur a assez d'argent
        if ($user['balance'] < $totalCost) {
            throw new Exception("Solde insuffisant pour effectuer cette
transaction.");
        }

        // 4. Mettre à jour le solde de l'utilisateur
        $stmt = $pdo->prepare("UPDATE users SET balance = balance - ? WHERE id =
?");
        $stmt->execute([$totalCost, $userId]);

        // 5. Mettre à jour ou créer une entrée dans le portefeuille
        $stmt = $pdo->prepare("
            SELECT id, quantity, avg_purchase_price FROM portfolio
            WHERE user_id = ? AND stock_id = ?
        ");
        $stmt->execute([$userId, $stockId]);
        $portfolio = $stmt->fetch();

        if ($portfolio) {
            // Calculer le nouveau prix d'achat moyen
            $totalShares = $portfolio['quantity'] + $quantity;
            $totalValue = ($portfolio['quantity'] *
$portfolio['avg_purchase_price']) + $totalCost;
            $newAvgPrice = $totalValue / $totalShares;

            // Mettre à jour l'entrée existante
            $stmt = $pdo->prepare("
                UPDATE portfolio
                SET quantity = quantity + ?, avg_purchase_price = ?
                WHERE id = ?
            ");
            $stmt->execute([$quantity, $newAvgPrice, $portfolio['id']]);
        } else {
            // Créer une nouvelle entrée
            $stmt = $pdo->prepare("
                INSERT INTO portfolio (user_id, stock_id, quantity,
avg_purchase_price)
                VALUES (?, ?, ?, ?)
            ");
            $stmt->execute([$userId, $stockId, $quantity,
$stock['current_price']]);
        }

        // 6. Enregistrer la transaction
        $stmt = $pdo->prepare("
            INSERT INTO transactions (user_id, stock_id, type, quantity, price,
total_amount)
            VALUES (?, ?, 'buy', ?, ?, ?)
        ");
        $stmt->execute([$userId, $stockId, $quantity, $stock['current_price'],
$totalCost]);

        // Valider la transaction
        $pdo->commit();
```

```
        return [
            'success' => true,
            'message' => "Achat réussi de $quantity actions pour un total de
$totalCost €"
        ];

    } catch (Exception $e) {
        // En cas d'erreur, annuler toutes les modifications
        $pdo->rollBack();
        return [
            'success' => false,
            'message' => $e->getMessage()
        ];
    }
}

// Fonction pour vendre des actions
function sellStock($userId, $stockId, $quantity, $pdo) {
    try {
        // Début de la transaction pour garantir l'intégrité des données
        $pdo->beginTransaction();

        // 1. Vérifier si l'utilisateur possède assez d'actions
        $stmt = $pdo->prepare("
            SELECT id, quantity, avg_purchase_price FROM portfolio
            WHERE user_id = ? AND stock_id = ?
        ");
        $stmt->execute([$userId, $stockId]);
        $portfolio = $stmt->fetch();

        if (!$portfolio || $portfolio['quantity'] < $quantity) {
            throw new Exception("Vous ne possédez pas assez d'actions pour
effectuer cette vente.");
        }

        // 2. Obtenir le prix actuel de l'action
        $stmt = $pdo->prepare("SELECT current_price FROM stocks WHERE id = ?");
        $stmt->execute([$stockId]);
        $stock = $stmt->fetch();

        // 3. Calculer le montant total de la vente
        $totalAmount = $stock['current_price'] * $quantity;

        // 4. Mettre à jour le solde de l'utilisateur
        $stmt = $pdo->prepare("UPDATE users SET balance = balance + ? WHERE id =
?");
        $stmt->execute([$totalAmount, $userId]);

        // 5. Mettre à jour le portefeuille
        if ($portfolio['quantity'] == $quantity) {
            // Supprimer l'entrée si toutes les actions sont vendues
            $stmt = $pdo->prepare("DELETE FROM portfolio WHERE id = ?");
            $stmt->execute([$portfolio['id']]);
        } else {
            // Mettre à jour la quantité
            $stmt = $pdo->prepare("
                UPDATE portfolio
```

```
                SET quantity = quantity - ?
                WHERE id = ?
            ");
            $stmt->execute([$quantity, $portfolio['id']]);
        }

        // 6. Enregistrer la transaction
        $stmt = $pdo->prepare("
            INSERT INTO transactions (user_id, stock_id, type, quantity, price,
total_amount)
            VALUES (?, ?, 'sell', ?, ?, ?)
        ");
        $stmt->execute([$userId, $stockId, $quantity, $stock['current_price'],
$totalAmount]);

        // Valider la transaction
        $pdo->commit();

        return [
            'success' => true,
            'message' => "Vente réussie de $quantity actions pour un total de
$totalAmount €"
        ];

    } catch (Exception $e) {
        // En cas d'erreur, annuler toutes les modifications
        $pdo->rollBack();
        return [
            'success' => false,
            'message' => $e->getMessage()
        ];
    }
}
?>
```

Données boursières simulées

Plutôt que d'utiliser une API en temps réel, utilisez des données simulées pour votre projet. Voici un exemple de fonction pour mettre à jour les prix des actions de façon aléatoire :

```
<?php
/**
 * Simule des variations de prix des actions pour un projet académique
 * Cette fonction peut être exécutée via une tâche cron toutes les heures
 */
function simulateStockPriceChanges($pdo) {
    try {
        // Récupérer toutes les actions
```

```php
        $stmt = $pdo->query("SELECT id, symbol, current_price FROM stocks");
        $stocks = $stmt->fetchAll();

        // Pour chaque action, simuler une variation de prix
        foreach ($stocks as $stock) {
            // Générer une variation aléatoire entre -3% et +3%
            $changePercent = mt_rand(-30, 30) / 10;

            // Calculer le nouveau prix
            $newPrice = $stock['current_price'] * (1 + ($changePercent / 100));

            // Arrondir à 2 décimales
            $newPrice = round($newPrice, 2);

            // Mettre à jour le prix dans la base de données
            $stmt = $pdo->prepare("
                UPDATE stocks
                SET current_price = ?,
                    last_updated = CURRENT_TIMESTAMP
                WHERE id = ?
            ");
            $stmt->execute([$newPrice, $stock['id']]);

            echo "Mise à jour du prix de {$stock['symbol']} :
{$stock['current_price']} € → {$newPrice} € ({$changePercent}%)\n";
        }

        return true;
    } catch (Exception $e) {
        echo "Erreur : " . $e->getMessage() . "\n";
        return false;
    }
}

// Connexion à la base de données
$host = 'localhost';
$db   = 'stockmanager';
$user = 'root';
$pass = '';
$charset = 'utf8mb4';

$dsn = "mysql:host=$host;dbname=$db;charset=$charset";
$options = [
    PDO::ATTR_ERRMODE            => PDO::ERRMODE_EXCEPTION,
    PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,
    PDO::ATTR_EMULATE_PREPARES   => false,
];

try {
    $pdo = new PDO($dsn, $user, $pass, $options);

    // Exécuter la simulation
    simulateStockPriceChanges($pdo);

} catch (\PDOException $e) {
    echo "Erreur de connexion : " . $e->getMessage() . "\n";
}
```

```
?>
```

Conseils pour la réalisation

1. **Utilisez un système de contrôle de version** comme Git pour collaborer efficacement. GitHub ou GitLab sont parfaits pour cela.
2. **Commencez par les fonctionnalités essentielles** et n'ajoutez des fonctionnalités supplémentaires que si vous avez du temps.
3. **Utilisez des frameworks et bibliothèques existants** pour accélérer le développement :
    - Bootstrap pour l'interface
    - Chart.js pour les graphiques
    - Un mini-framework PHP comme Slim ou un template de base MVC
4. **Simplifiez vos données** en utilisant des données statiques ou simulées au lieu d'API en temps réel qui ajoutent de la complexité.
5. **Réunissez-vous régulièrement** (tous les 2-3 jours) pour faire le point sur l'avancement et ajuster les tâches si nécessaire.
6. **Prévoyez du temps pour les tests** afin d'identifier et corriger les bugs avant la remise du projet.

Cette approche simplifiée vous permettra de livrer un projet fonctionnel en 2-3 semaines, tout en respectant les contraintes de compétences et de temps de votre équipe. Si vous avez d'autres questions sur des aspects spécifiques du développement, n'hésitez pas à me demander !