GoalTeller takehome assignment
Frontend Intern

Introduction

This assignment evaluates your React, TypeScript, CSS, and responsive design skills in a fintech context. You will build a Portfolio Summary Component that displays user asset data by integrating with a mock API and adapts across screen sizes.

Mock Data API

Fetch your data from the following mock API endpoint (no hard-coding):

- https://raw.githubusercontent.com/aashishsingla567/gt-takehome/refs/heads/main/portfolios.json

This endpoint returns a JSON array of objects in the shape:

```
[
  {
    "symbol": "AAPL",
    "quantity": 10,
    "avgPrice": 120.50,
    "currentPrice": 145.30
  },
  {
    "symbol": "GOOG",
    "quantity": 5,
    "avgPrice": 1500.00,
    "currentPrice": 1520.75
  },
  {
    "symbol": "TSLA",
    "quantity": 3,
    "avgPrice": 620.00,
    "currentPrice": 610.20
  }
]
```

Task

Using the data from the API above, implement a React component in TypeScript that:

1. Displays a table or grid of assets with columns:

   - **Symbol**
   - **Quantity**
   - **Avg. Price**

- **Current Price**
- **Total Value** (quantity × currentPrice)
- **P/L (%)** $((currentPrice - avgPrice)/avgPrice) \times 100$

2. Allows sorting by clicking any column header.

3. Provides a search input to filter assets by symbol (case-insensitive).

4. **Responsive Design is mandatory**: use CSS Grid or Flexbox to switch from a multi-column desktop view to a single-column mobile view.

5. Style using CSS Modules or styled-components:

   - Use CSS custom properties for primary and secondary colors.
   - Add a hover effect on rows (e.g., background-color change).

6. Ensure accessibility: semantic HTML, ARIA roles for the table, and keyboard focus styles.

Bonus Task (Optional)

Add a bar chart showing asset allocation percentages using a chart library (e.g., Recharts). Place the chart above or beside the asset list, responsive to different screen sizes.

Deliverables

- A GitHub repository or ZIP containing:

  - `/src` with React + TypeScript code and styles.
  - `README.md` with setup and run instructions.

- Optional: test files for sorting/search utilities.

Evaluation Criteria

- **React & TypeScript:** Proper types, props, state, and hooks.
- **Core JavaScript:** Correct API integration, calculations, sorting, filtering logic.
- **CSS:** Responsive layout, custom properties, and hover effects.
- **Accessibility:** ARIA roles, keyboard navigation, and color contrast.
- **Code Quality:** Structure, readability, and documentation.
- **Bonus:** Functional and responsive chart integration.

Timeline

Please submit your assignment within 7 days. Email your repository link to the hiring team. Good luck!