ADBERT 前端工程師面試題目__林昱均

這是一個使用 React + Material-UI + Redux + Recharts 實現的前端面試項目，並遵循業界最佳實踐進行了全面重構。

🎯 項目概述

本項目包含兩個主要功能模塊：

第一題：Material-UI ButtonGroup 操作邏輯

- 使用 Material-UI 的 `ButtonGroup` 組件，並實現垂直佈局。
- 實現計數器功能（CLICK 按鈕）。
- 實現清除功能（CLEAR 按鈕）。
- 實現禁用/啟用功能（DISABLE/ABLE 按鈕）。
- 使用 **Redux Toolkit** 進行狀態管理。
- 遵循 Material-UI 最佳實踐，全面使用 `sx` 屬性進行樣式管理。

第二題：串接公開 API 畫圖表

- 使用 **Recharts** 圖表庫。
- 串接 **Frankfurter.app** 公開 API，獲取真實歷史匯率數據。
- 實現一個**堆疊長條圖 (Stacked Bar Chart)**，展示 EUR 對 GBP 和 USD 的匯率變化。
- 包含圖例 (Legend) 和工具提示 (Tooltip)，並進行了細節微調。

🚀 快速開始

安裝依賴

```
npm install
```

啟動開發服務器

```
npm start
```

應用將在 http://localhost:3000 啟動。

構建生產版本

```
npm run build
```

📦 技術棧

- **React 18** - 前端框架
- **Material-UI (MUI) 5** - UI 組件庫
- **Redux Toolkit** - 狀態管理
- **Recharts** - 圖表庫
- **Frankfurter.app API** - 公開匯率 API 數據源

🎨 功能特色

- ✅ **遵循最佳實踐**: 全專案使用 Material-UI 的 `sx` 屬性和語義化組件 (`<Box>`, `<Typography>`) 進行樣式管理，而非內聯 `style` 或 `styled-components`，確保了程式碼的高度一致性與可維護性。
- ✅ **Redux 狀態管理**: 使用 Redux Toolkit 高效管理第一題的按鈕狀態，展現了複雜應用的狀態管理能力。
- ✅ **真實 API 串接**: 第二題串接了真實的 Frankfurter.app 公開 API，動態獲取並展示了歷史匯率數據。
- ✅ **精緻的數據可視化**: 使用 Recharts 實現了符合設計稿的堆疊長條圖，並根據使用者回饋進行了細節微調（如柱體寬度、圖例間距）。
- ✅ **完整的用戶體驗**: 包含了響應式設計、API 請求的錯誤處理和載入狀態，確保了應用的穩定性。

📁 項目結構

```
src/
├── components/
│   ├── ButtonGroup.js      # 第一題：按鈕組件
│   └── ChartComponent.js   # 第二題：圖表組件
├── store/
│   ├── counterSlice.js     # Redux slice
│   └── store.js            # Redux store 配置
├── App.js                  # 主應用組件
└── index.js                # 應用入口
```

📝 部署說明

本地部署

1. 克隆項目
2. 安裝依賴：`npm install`
3. 啟動開發服務器：`npm start`

雲端部署 (Vercel)

1. 將代碼推送到 GitHub
2. 在 Vercel 中導入 GitHub 倉庫
3. 自動部署完成

🔧 自定義配置

可以在 `src/App.js` 中修改 Material-UI 主題配置，或在各個組件中透過 `sx` 屬性調整樣式。

📞 聯繫方式

如有任何問題或建議，請隨時聯繫：林昱均 erica51528@gmail.com。