Lakshmi Manasa Pothakamuru, **Full Stack Developer Intern role at Blockhouse**

Chart Dashboard Application

This application is a chart dashboard that displays various types of charts (Candlestick, Line, Bar, and Pie charts) using data provided by a backend server. The frontend is built with React and Chart.js, while the backend, developed with Django, serves hardcoded data for each chart type.

Table of Contents

Setup Instructions

Backend Setup (Windows)

1. Clone the repository: **https://github.com/Manasapothakamuru/BlockHouse.git**

   Open Command Prompt and run:
   ```
   git clone <repository-url>
   cd <repository-directory>\Backend
   ```

2. Create a virtual environment and activate it:

   Run the following commands:
   python -m venv env
   .\env\Scripts\activate

3. Install the required packages:

   Install Django and other dependencies by running: **pip install django djangorestframework** django-cors-headers

4. Run the Django development server: **python manage.py runserver**
   The backend server will start running at **`http://127.0.0.1:8000/`**.

5. API Endpoints:

- `GET /api/candlestick-data/` - Fetches candlestick chart data.
- `GET /api/line-chart-data/` - Fetches line chart data.
- `GET /api/bar-chart-data/` - Fetches bar chart data.
- `GET /api/pie-chart-data/` - Fetches pie chart data.

Frontend Setup (Windows)

1. Navigate to the frontend directory: cd ..\Frontend

2. Install the required packages:

Install Node.js packages by running: **npm install react react-dom next chart.js react-chartjs-2 axios**

3. Run the React development server: **npm run dev**

The frontend application will start running at **`http://localhost:3000/`.**

Libraries and Tools Used

Backend
- Django: A high-level Python web framework for rapid development and clean, pragmatic design.
- Django REST Framework: A powerful and flexible toolkit for building Web APIs.
- CORS Headers: Django application to enable Cross-Origin Resource Sharing (CORS) for AJAX requests from different domains.

Frontend
- React: A JavaScript library for building user interfaces.
- Next.js: A React framework for server-side rendering and static site generation.
- Chart.js: A simple yet flexible JavaScript charting library for visualizing data.
- React-Chartjs-2: React wrapper for Chart.js.
- Axios: A promise-based HTTP client for the browser and Node.js to make API requests.

Approach and Thought Process

Backend
The backend uses Django to create a REST API that provides data for different types of charts. Each API endpoint sends data in a JSON format that is specifically structured for a particular chart type. Django REST Framework is used because it is easy to work with and works well with Django's database system, making development faster and data management more efficient.

Frontend

The frontend is built using React to create a dynamic and responsive user interface. It uses Chart.js, through the react-chartjs-2 library, to display different types of charts like Candlestick, Line, Bar, and Pie charts. The data for these charts is fetched from the backend using Axios, which allows it to be done smoothly in the background. The app is designed to handle data fetching efficiently by using React hooks like useEffect and useState to manage data and keep everything up to date.

CandlestickChart: Displays financial data with candlesticks for each day, showing open, close, high, and low prices. It uses 7 days of data to track price changes over time.

LineChart: Visualizes time-series data with a line connecting monthly data points ("Jan" to "Apr") and their values (10 to 40), helping to identify trends and patterns.

BarChart: Compares values across categories, such as sales for "Product A", "Product B", and "Product C" with values (100, 150, 200), making category comparisons easy.

PieChart: Illustrates proportions of a whole with segments ("Red", "Blue", "Yellow") and their values (300, 50, 100), showing how each part contributes to the total.

Output: