Created by: Muhammad Hussain

CryptoPulse Development Documentation

Project Overview

CryptoPulse is a real-time cryptocurrency dashboard built with React, featuring live price tracking, advanced filtering, and interactive charts powered by the CoinGecko API.

This document demonstrates the approach taken, combining independent problem-solving with targeted AI assistance for specific challenges faced during implementation.

Self-Implemented Components

Project and Libraries Setup:

```
npm create vite@latest cryptopulse -- --template react
cd cryptopulse
npm install
npm install @headlessui/react chart.js react-chartjs-2
npm install -D tailwindcss postcss autoprefixer
```

Core State Management Logic:

```
const [page, setPage] = useState(1);
const [loadingMore, setLoadingMore] = useState(false);
const [hasMore, setHasMore] = useState(true);

// Advanced filtering system
```

```
const [sortBy, setSortBy] = useState('market_cap');
const [sortOrder, setSortOrder] = useState('desc');
const [priceRange, setPriceRange] = useState({ min: '',
max: '' });
const [searchTerm, setSearchTerm] = useState('');

// Modal state management
const [isModalOpen, setIsModalOpen] = useState(false);
const [selectedCoin, setSelectedCoin] = useState(null);
```

API Integration:

```
const fetchData = useCallback(async () => {
    // Global stats endpoint
    const globalRes = await
fetch('https://api.coingecko.com/api/v3/global');

    const coinsRes = await fetch(
"https://api.coingecko.com/api/v3/coins/markets?vs_currency
=usd&order=market_cap_desc&per_page=100&page=${page}&sparkl
ine=false&price_change_percentage=1h%2C24h%2C7d");

    const vanryRes = await fetch(
"https://api.coingecko.com/api/v3/coins/markets?vs_currency
=usd&ids=vanar-chain&price_change_percentage=1h%2C24h%2C7d"
);
}, [page]);
```

Data Preprocessing Logic (Searching, Filtering):

```
useEffect(() => {
    let dataToProcess = [...allCryptoData];
```

```
    // Search implementation
    if (searchTerm) {
dataToProcess = dataToProcess.filter(crypto =>

crypto.name.toLowerCase().includes(searchTerm.toLowerCase()
) ||

crypto.symbol.toLowerCase().includes(searchTerm.toLowerCase
())
        );
    }
    // Price range filtering
    const min = parseFloat(priceRange.min);
    const max = parseFloat(priceRange.max);
    if (!isNaN(min) && min >= 0) {
        dataToProcess = dataToProcess.filter(c =>
c.current_price >= min);
    }
    if (!isNaN(max) && max >= 0) {
        dataToProcess = dataToProcess.filter(c =>
c.current_price <= max);
    }

    // Dynamic sorting system
    const sortKeyMap = {
        'market_cap': 'market_cap',
        'price': 'current_price',
        'change24h': 'price_change_percentage_24h'
    };

    const sortKey = sortKeyMap[sortBy];
    if (sortKey) {
```

```
        dataToProcess.sort((a, b) => {
            const valA = a[sortKey] || -Infinity;
            const valB = b[sortKey] || -Infinity;
            return sortOrder === 'asc' ? valA - valB : valB
- valA;
        });
    }
    setFilteredData(dataToProcess);
}, [searchTerm, allCryptoData, sortBy, sortOrder,
priceRange]);
```

Business Logic:

```
// Intelligent number formatting
const formatCurrency = (amount) => {
    if (amount === null || amount === undefined) return
'N/A';
    return new Intl.NumberFormat('en-US', {
        style: 'currency',
        currency: 'USD',
        minimumFractionDigits: amount < 1 ? 4 : 2,
        maximumFractionDigits: amount < 1 ? 8 : 2
    }).format(amount);
};

const formatLargeNumber = (num) => {
    if (num === null || num === undefined) return 'N/A';
    if (num >= 1e12) return (num / 1e12).toFixed(2) + 'T';
    if (num >= 1e9) return (num / 1e9).toFixed(2) + 'B';
    if (num >= 1e6) return (num / 1e6).toFixed(2) + 'M';
    return num.toLocaleString();
};
```

```
// Event handlers and user interaction logic
const handleLoadMore = () => {
    if (!loadingMore && hasMore) {
        setPage(prevPage => prevPage + 1);
    }
};

const handleRefresh = () => {
    if (page === 1) {
        fetchData();
    } else {
        setPage(1);
    }
    setHasMore(true);
    setSearchTerm('');
    setPriceRange({ min: '', max: '' });
};
```

Responsive-Design Implementation:

```
// Grid responsiveness
className="grid grid-cols-1 md:grid-cols-2 xl:grid-cols-3
gap-10"
className="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-4
gap-8"

// Flexbox responsiveness
className="flex flex-col sm:flex-row justify-center
items-center gap-4"
```

Dark/Light mode toggle implementation:

```
// Theme state management
const [theme, setTheme] = useState('light');

// Auto-detect user's system preference and load saved
theme
useEffect(() => {
    const savedTheme = localStorage.getItem('theme');
    const userPrefersDark =
window.matchMedia('(prefers-color-scheme: dark)').matches;
    if (savedTheme) {
        setTheme(savedTheme);
    } else if (userPrefersDark) {
        setTheme('dark');
    }
}, []);

// Apply theme changes to document and persist to
localStorage
useEffect(() => {
    const root = window.document.documentElement;
    root.classList.remove('light', 'dark');
    root.classList.add(theme);
    localStorage.setItem('theme', theme);
}, [theme]);

// Theme toggle functionality
const toggleTheme = () => {
    setTheme(prevTheme => (prevTheme === 'light' ? 'dark' :
'light'));
};
```

```
// Theme toggle button in header
<button
    onClick={toggleTheme}
    className="p-3 bg-white/15 dark:bg-gray-800/50
rounded-full text-yellow-300 backdrop-blur-lg shadow-lg
transition-colors hover:bg-white/25
dark:hover:bg-gray-700/60"
>
    {theme === 'light' ? (
        <i className="fas fa-moon text-xl"></i>
    ) : (
        <i className="fas fa-sun text-xl"></i>
    )}
</button>
```

AI-Assisted Implementation:

UI/UX Design and Layout:

Prompt:

“Design a modern cryptocurrency dashboard with glassmorphism effects. I want cards for each crypto, a gradient background, and a professional look with good spacing and typography.”

Chart.js Integration:

Prompt:

“Integrate Chart.js with React for cryptocurrency price charts. Need line charts with time period switching including 1D,7D,30D and 1Y.”

```
ChartJS.register(CategoryScale, LinearScale, PointElement,
LineElement, Title, Tooltip, Legend, Filler);

// Time period selector buttons
{[1, 7, 30, 90, 365].map(d => (
    <button
        key={d}
        onClick={() => setDays(d)}
        className="px-3 py-1 rounded-md text-sm
font-semibold"
    >
        {d === 1 ? '24H' : d === 365 ? '1Y' : `${d}D`}
    </button>
))}

// Chart data formatting
const formattedData = {
    labels: data.prices.map(price => new
Date(price[0]).toLocaleDateString()),
    datasets: [{
        label: `Price (USD)`,
        data: data.prices.map(price => price[1]),
        borderColor: '#8b5cf6',
        backgroundColor: 'rgba(139, 92, 246, 0.1)',
        tension: 0.1,
        fill: true,
    }],
};
```

Modal Implementation with Headless UI:

Prompt:

“Create a modal dialog box using Headless UI React for every cryptocurrency displaying coin information and integration of chart js upon clicking the coin box.”

```
<Dialog.Panel className="w-full max-w-3xl rounded-3xl
bg-white p-6 md:p-8">
  {/* Header with coin info */}
  <div className="flex flex-col sm:flex-row items-start
sm:items-center gap-4 mb-6">
    <img src={coin.image} alt={coin.name} className="w-16
h-16" />
    <Dialog.Title className="text-3xl
font-bold">{coin.name}</Dialog.Title>
    <p className="text-lg text-gray-500
uppercase">{coin.symbol}</p>
  </div>

  {/* Chart section */}
  <div className="mb-6">
    <h3 className="text-xl font-semibold mb-2">Price
Chart</h3>
    {/* Time period buttons */}
    <div className="flex flex-wrap gap-2 mb-4">
      {[1, 7, 30, 90, 365].map(d => (...))}
    </div>
    {/* Interactive chart */}
    <div className="h-64">
      <Line data={chartData} options={...} />
    </div>
  </div>
</Dialog.Panel>
```