Project Title: `Crypto-Portfolio` App

Objective:

Develop a single-page application (SPA) named, where users can:

1. Add tokens to their watch list.
2. View their current balance for each token.
3. View the historical balance of each token based on date.
4. Check their token allowance.
5. Perform operations on the token, ex: transfer to another address, approve token.

Requirements:

1. **Wallet Connection:**
    - Users should be able to connect their own Metamask or any other wallet.
    - Alternatively, users should be able to provide a wallet address as an input.
2. **Watch List:**
    - Users can add various tokens to their watch list.
    - Display the current balance of each token in the watch list.
3. **Historical Data:**
    - Fetch and display the historical balance of each token.
    - Provide a date picker for users to select the date range.
4. **Allowance:**
    - Users should be able to check their token allowance for different smart contracts.
5. **Token Transfer:**
    - Implement a functionality that allows users to transfer tokens to another address.
    - Include form fields for the recipient address and amount to be transferred.
6. **Visual Representations:**
    - Use tables, charts, and graphs to represent token balances, historical data, and allowances.
    - Be as creative as possible with the visual representation of data.

Deliverables:

1. **GitHub Repository:**
    - Create a GitHub repository with an appropriate name for your project.
    - Do not add all your code in a single commit! Follow Github's bestPractices.

    - Ensure the repository is well-organized and includes a README file with clear instructions on how to run the project.
2. **Working Demo (Bonus):**
    - Deploy your application on Netlify (or any other hosting service).
    - Provide the link to the live demo in your README file.
3. **Visual Representations:**
    - Include as many visual representations as possible.
    - Use tables and graphs to enhance the user experience and make the data easy to understand.

Technical Stack:

- **Frontend:** React.js or any other modern frontend framework.
- **Blockchain Interaction:** Web3.js, Ethers.js, or any other library to interact with the Ethereum blockchain.
- **Backend (optional):** Node.js, Express, or any other backend framework (if required for your implementation).
- **Database (optional):** MongoDB, Firebase, or any other database (if required for your implementation).

Evaluation Criteria:

1. **Functionality:**
    - The application should meet all the specified requirements.
    - The wallet connection, token watch list, allowance check, and transfer features should work seamlessly.
2. **Code Quality:**
    - The code should be clean, well-organized, and properly commented.
    - Follow best practices for coding standards and conventions.
3. **User Interface:**
    - The application should be visually appealing and user-friendly.
    - Creative and effective use of tables, charts, and graphs.
4. **Documentation:**
    - A comprehensive README file that includes:
        - Project overview.
        - Instructions on how to set up and run the project.
        - Description of the features.
        - Link to the working demo (if deployed).

Note:

1. Plagiarism is not tolerated.

2. Using ChatGPT/Online examples or similar tools is allowed, but ensure originality in your code and approach.
    1. If you are using any inspiration/reference from examples. Just add the related references in the relevant places.
3. Your attempt and effort to solve the problem will be valued more than a perfect solution.

Good luck, and happy coding!