Cleo hiring chart task

Overview

For this assignment, you will be extending a preexisting web application. The provided app is a very simple candlestick chart. There are two features missing - candle loader and information about selected candle. The goal of this assignment is to submit runnable solution with these two features implemented. Your solution code should pass linter check and type check. You should not add new dependencies or modify project configuration unless there is a very good reason to do so.

Specification

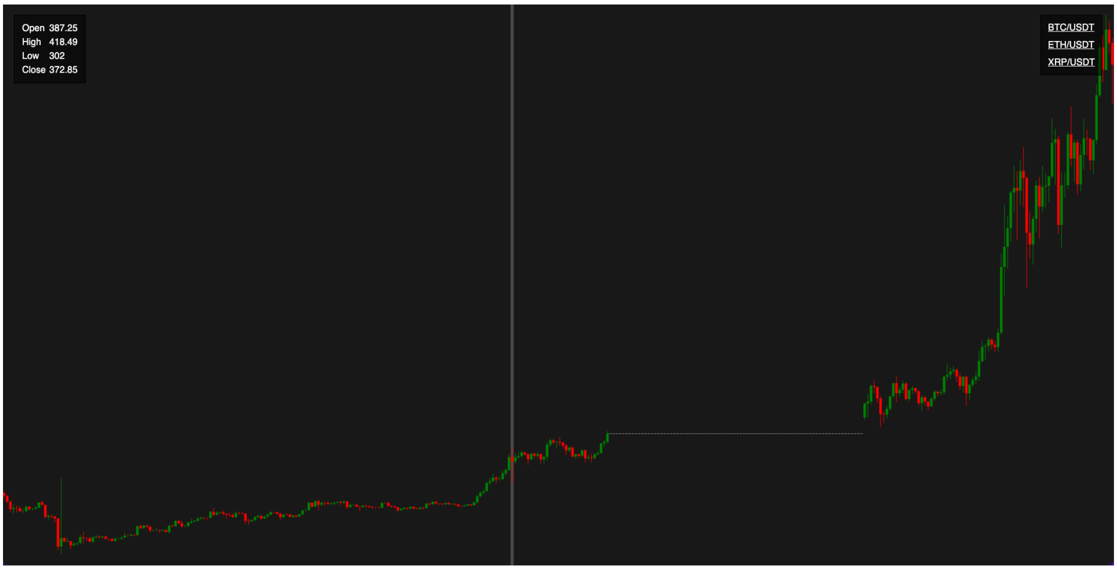
CANDLE LOADER
Candlestick data used by the chart must be fetched from a server. The data is served from **public/candles** directory with several prepared examples. The candles are in flattened array structure. Each candle is composed of five different numbers in the following order: open price, high price, low price, close price and volume; volume is nullable. Volume is not visible in the chart and should be skipped while parsing the data. Data gaps may occur, empty candles are represented as single null value. Null candles at the start of the array must be ignored, null candles anywhere else must be transformed into candles with open, high, low and close price equal to previous close price.

Example:
[null, 20, 30, 10, 25, null, null]
=> [{ open: 20, high: 30, low: 10, close: 25 }, { open: 25, high: 25, low: 25, close: 25 }]

Implement logic of **useCandleLoader** module according to specification and with use of pre-existing type definitions.

SELECTED CANDLE INFO
Box in the top left corner must show information about selected candle. Candle is selected when mouse cursor hovers over the candle or anywhere above/below it. Selected candle should be visually highlighted. The highlight may look like on the screenshot below.

Extend logic of **Chart** module according to specification and pass the candle to CandleInfo component inside of **App** module.



Open 387.25
High 418.49
Low 302
Close 372.85

Guidelines

- This task is designed to take around 2 hours for an experienced developer
- Focus on quality rather than completing as much as possible
- Add comments to address issues you didn't have time to handle

Recommendations

- Use IDE with ESLint and Prettier extensions installed and configured
    - Make sure lint and check-types scripts pass before submitting
- Avoid adding new dependencies or changing existing dependencies
- Write type-safe code and follow provided types
    - E.g.: don't use *any* type unless there is no other way
- Explain any assumptions made in comments
- Do **NOT** include unnecessary files in your submission
- Do **NOT** disable ESLint rules to silence valid warnings/errors
- Do **NOT** modify existing code in order to make assignment easier

Evaluation Criteria

- Adherence to specification
- Functionality in all scenarios
- Readability, clean code and code structure

Submission

- Deadline for submission is **7 days** from the day of receiving the task
- Sending as a zip or uploading to the public git repository is fine
- Fill up the Summary.txt