NatWest Frontend Code Test

Thank you for taking the time to complete this code test. We appreciate you taking the time to complete this test, hopefully you'll find it interesting too. There is no time limit, but it shouldn't take you longer than a couple of hours.

There is no right or wrong answer, we are more interested in how you approach the problem, your design considerations, and the best practices you follow.

Instructions

Stub API and Shell React App code download location: https://github.com/Frontend-Interview-Test/test/raw/main/Natwest%20Frontend%20Code%20Test.zip

We would like you to create a React application to fetch and display payments from an API. find the stub API running locally at 'http://localhost:9001/api' when running the app using 'npm start' and a schema for the API can be found below.

GET /Payments

```
{
    metaData: {
        ...
    },
    results: [
        {
            paymentAmount: "120.00",
            paymentCurrency: "GBP",
            paymentType: "CHAPS",
            paymentDate: "15-Jun-2020",
            paymentStatus: "W", // ( A=Approved, C=Cancelled, P=Pending Approval)
            toAccount: {
                accountName: "",
                sortCode: "341234",
                accountNumber: "125365",
            },
            fromAccount: {
                accountName: "",
                sortCode: "341234",
                accountNumber: "125365",
            },
        },
        ...
    ],
};
```

We have included a shell React app to get you started or you can setup your own app if you wish. Feel free to use React state or Redux but think about how you structure your app and share data between components. You can also use any libraries that you want to help you.

We don't mind how the app looks; a simple table is fine. We have included Bootstrap if you would like to use it. You can find the documentation for bootstrap here - https://react-bootstrap.github.io/components/alerts/

Task 1

Fetch and display the payments using the above stub API.

Task 2

There can be multiple pages of payments. In the response you will find some metadata as shown below. Use this metadata to paginate through the payments. This can be done with a load more button or when scrolling to the bottom of the list.

GET /payments?pageIndex=kdDhsa93h

```
{
    metaData: {
        hasMoreElements: true,
        nextPageIndex: 'kJne893j'
    },
    results: [
        {
            paymentAmount: "120.00",
            paymentCurrency: "GBP",
            paymentType: "CHAPS",
            paymentDate: "15-Jun-2020",
            paymentStatus: "W", // ( A=Approved, C=Cancelled, P=Pending Approval)
            toAccount: {
                accountName: "",
                sortCode: "341234",
                accountNumber: "125365",
            },
            fromAccount: {
                accountName: "",
                sortCode: "341234",
                accountNumber: "125365",
            },
        },
        ...
    ],
};
```

Task 3

Add a filter to the frontend to allow the user to view only payments that are pending approval

Deliverables

- Project containing source code for your solution.
- Any instructions to run your app.

- A short description of what you have implemented, include any info about how you would improve your solution if you were to spend more time on it

Please check-in your code to a GitHub repository, add a detailed readme file and provide details and access.