Implementing the Issues Feature: Server-Side Filter

Here we briefly explain how to implement server-side filtering.

WE'LL COVER THE FOLLOWING

- Exercises

Before starting with the server-side filtering, let's recap the last lesson in case you had difficulties with it. Basically, you can perform the refactoring in three steps. First, we need the following package: `recompose` which is already installed in our environment.

Secondly, import the `withState` higher-order component in the *src/Issue/IssueList/index.js* file and use it to wrap our exported `Issues` component, where the first argument is the property name in the local state, the second argument is the handler to change the property in the local state, and the third argument is the initial state for that property.

Environment Variables

Key:	Value:
REACT_APP_GITHUB...	Not Specified...
GITHUB_PERSONAL...	Not Specified...

```
import React from 'react';
import { Query } from 'react-apollo';
import gql from 'graphql-tag';
import { withState } from 'recompose';

...

export default withState(
  'issueState',
  'onChangeIssueState',
  ISSUE_STATES.NONE,
)(Issues);
```

src/Issue/IssueList/index.js

Finally, refactor the `Issues` component from a class component to a functional stateless component. It accesses the `issueState` and `onChangeIssueState()` function in its props now. Remember to change the usage of the `onChangeIssueState` prop to being a function and not a class method anymore.

Environment Variables

Key:	Value:
REACT_APP_GITHUB...	Not Specified...
GITHUB_PERSONAL...	Not Specified...

```
...

const Issues = ({
  repositoryOwner,
  repositoryName,
  issueState,
  onChangeIssueState,
}) => (
  <div className="Issues">
    <ButtonUnobtrusive
      onClick={() => onChangeIssueState(TRANSITION_STATE[issueState])}
    >
      {TRANSITION_LABELS[issueState]}
    </ButtonUnobtrusive>

    ...
  </div>
);

...
```

src/Issue/IssueList/index.js

The previous lesson makes writing stateful components easier, where the state is much more convenient. Next, we advance the filtering from the client-side to the server-side. We use the defined GraphQL query and its arguments to make a more exact query by requesting only open or closed issues. In the *src/Issue/IssueList/index.js* file, extend the query with a variable to specify the issue state:

Environment Variables

Key:	Value:

REACT_APP_GITHUB...	Not Specified...

```
const GET_ISSUES_OF_REPOSITORY = gql`
  query(
    $repositoryOwner: String!
    $repositoryName: String!
    $issueState: IssueState!
  ) {
    repository(name: $repositoryName, owner: $repositoryOwner) {
      issues(first: 5, states: [$issueState]) {
        edges {
          node {
            id
            number
            state
            title
            url
            bodyHTML
          }
        }
      }
    }
  }
`;
```

src/Issue/IssueList/index.js

Next, we use the `issueState` property as a variable for our `Query` component. In addition, we remove the client-side filter logic from the `Query` component's render prop function.

Environment Variables

Key:	Value:
REACT_APP_GITHUB...	Not Specified...
GITHUB_PERSONAL...	Not Specified...

```
const Issues = ({
  repositoryOwner,
  repositoryName,
  issueState,
  onChangeIssueState,
}) => (
  <div className="Issues">
    ...

    {isShow(issueState) && (
      <Query
        query={GET_ISSUES_OF_REPOSITORY}
        variables={{
          repositoryOwner,
          repositoryName,
          issueState,
        }}
      >
        {({ data, loading, error }) => {
          if (error) {
            return <ErrorMessage error={error} />;
          }

          const { repository } = data;

          if (loading && !repository) {
            return <Loading />;
          }

          return <IssueList issues={repository.issues} />;
        }}
      </Query>
    )}
  </div>
);
```

src/Issue/IssueList/index.js

We are now only querying open or closed issues. Our query became more exact, and the filtering is no longer handled by the client.

We have also implemented the pagination feature for the Issue features. Check it out:

Environment Variables

Key:	Value:
REACT_APP_GITHUB...	Not Specified...
GITHUB_PERSONAL...	Not Specified...

```
import React from 'react';

import Link from '../../Link';

import './style.css';

const Footer = () => (
  <div className="Footer">
    <div>
      <small>
        <span className="Footer-text">Built by</span>{' '}
        <Link
          className="Footer-link"
          href="https://www.robinwieruch.de"
        >
          Robin Wieruch
        </Link>{' '}
        <span className="Footer-text">with &hearts;</span>
      </small>
    </div>
    <div>
      <small>
        <span className="Footer-text">

          Interested in GraphQL, Apollo and React?
        </span>{' '}
        <Link
          className="Footer-link"
          href="https://www.getrevue.co/profile/rwieruch"
        >
          Get updates
        </Link>{' '}
        <span className="Footer-text">
          about upcoming articles, books &
        </span>{' '}
        <Link className="Footer-link" href="https://roadtoreact.com">
          courses
        </Link>
        <span className="Footer-text">.</span>
      </small>
    </div>
  </div>
);

export default Footer;
```

Exercises

1. Confirm your source code for the last section