Instruction for test writing

You should never use real data from the server for testing. Use only mocked data.

Reducer testing

Reducer - is a function, that takes state and action and returns new state, so you can test it like general function.

- Go to the link. Find the direct example for test `should return the initial state`.

> You don't need to install `Babel`

- Use this example for writing your tests.

Components testing

To test the React component you need to render this component in the test.
Use link to see example of rendering.
If you use store data in testing component you need to cover component in the `Provider` and pass the `mocked store`.

Example of mockedStore:

```
const mockedState = {
  user: {
    isAuth: true,
    name: 'Test Name',
  },
  courses: [],
  authors: [],
};

const mockedStore = {
    getState: () => mockedState,
    subscribe: jest.fn(),
    dispatch: jest.fn(),
};
```

Example of using mockedStore in the test:

```
render(
  <Provider store={mockedStore}>
    <Header />
  </Provider>
);
```

Use `.toBeInTheDocument()` to check if something is on the page.

Example of expectation

```
expect(screen.queryByText('Test Name')).toBeInTheDocument();
```

- Use this link to see different queries.
- To simulate events *('click' for buttons and 'change' for input)* you can use Firing Events.
- Read "How To Mock Fetch in Jest" article to learn how to mock fetch.