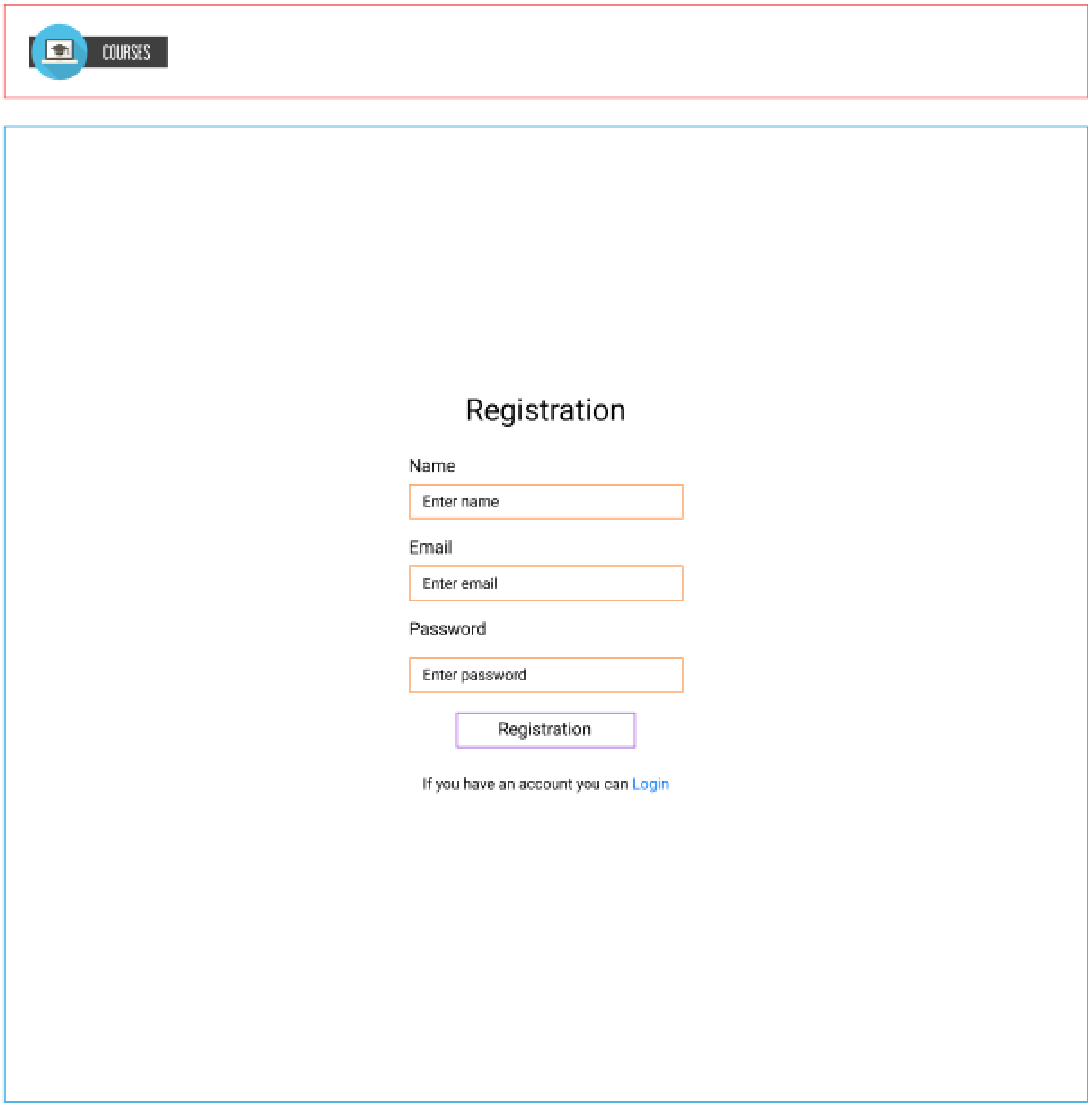
Description

App component

Add the router to the App component.

All movements on the pages of the application can be carried out through the react-router-dom.
Router should be described in `App.jsx`.

Registration (new component)



(example)

Create `Registration` component.

- This component should contain the following elements:

- `name` field;
- `email` field;
- `password` fields;
- `Regisatration` button;
- link that navigates to `Login` component.

- This component should be rendered by route `/registration`.
  `Registration` component should have functionality to send request to API for creating new user. See `/register` endpoint in API Swagger.
- After successful registration, user is redirected to the `Login` page by route `/login`.
- Use `react-router-dom` hook `useHistory` to redirect from one url to another.
- Use `form` tag.
- You should reuse `Input` and `Button` components.
- Request should be sent by `submit` event. Use `onSubmit` props for `form`.

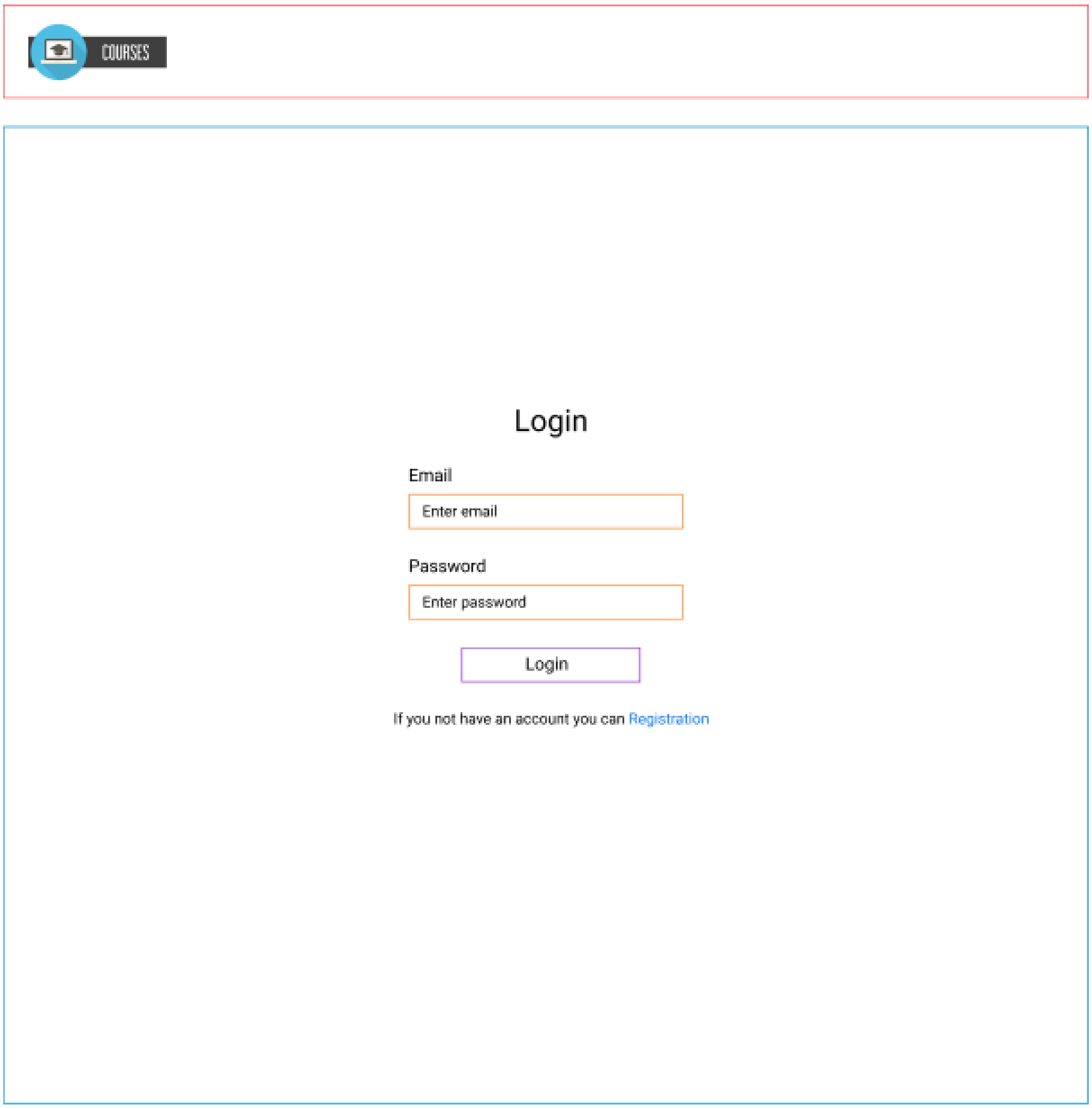
Example of POST request using `fetch`:

```
const newUser = {
  name,
  password,
  email,
};

const response = await fetch('http://localhost:3000/register', {
    method: 'POST',
    body: JSON.stringify(newUser),
    headers: {
      'Content-Type': 'application/json',
    },
});

const result = await response.json();
```

Login (new component)



Login

Email

Enter email

Password

Enter password

Login

If you not have an account you can Registration

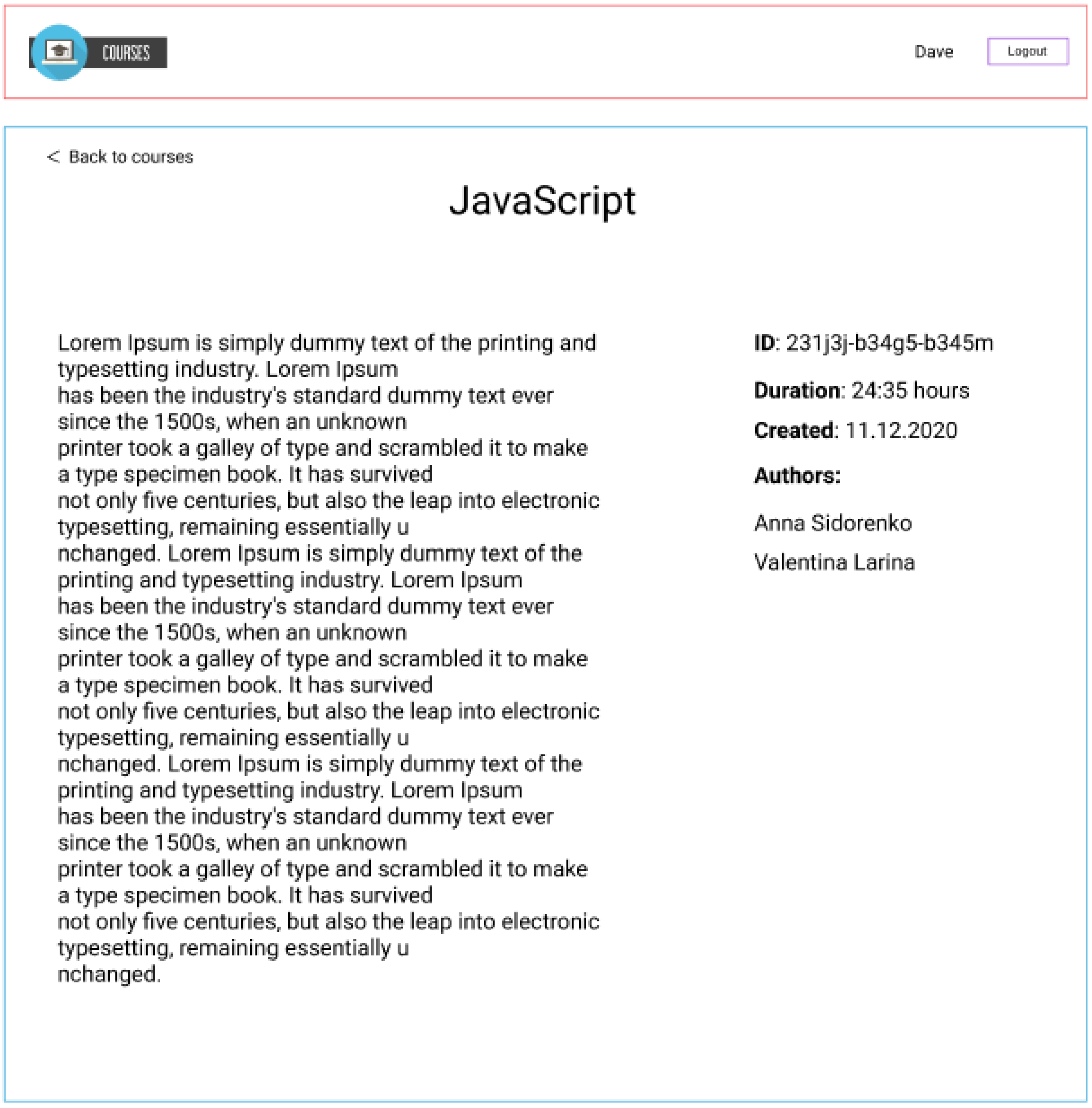
(example)

Create `Login` component.

- This component should contain the following elements:
  - `email` field;
  - `password` fields;
  - `Login` button;
  - Link to `Regisatration` component (use `Link` from `react-router-dom`).
- `Login` should be rendered by route `/login`;
- `Login` should have functionality that send request to API for getting token.
  See `/login` endpoint in API Swagger.
- Use `form` tag.

- You should use `Input` and `Button` components.
- Request should be sent by `submit` event. Use `onSubmit` props for `form`.
- Response contains value `result`, it's user's token. You should save it to the `localStorage`.
- After successful registration, user is redirected to the `Courses` page by route `/courses`.
- Use `react-router-dom` hook `useHistory` to redirect from one url to another

Course info (new component)



JavaScript

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially u nchanged. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially u nchanged. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially u nchanged.

ID: 231j3j-b34g5-b345m

Duration: 24:35 hours

Created: 11.12.2020

Authors:

Anna Sidorenko

Valentina Larina

(example)

Create `CourseInfo` component.

- This component shows information about course.
- This component contains:

    - ID of course;
    - Title;
    - Description;
    - Duration;
    - list of authors;
    - Creation date;
    - `Back to courses` button.

- This component should be rendered by route `/courses/:courseId`.
- Find course in courses list by an id using `courseId` path-param.
- Use `react-router-dom` hook `useParams` to get courseId from url.
- `Back to course` button should navigate to `/courses` and then the `Courses` component appears.
  Use Link from `react-router-dom`.

Courses

Implement new features for `Courses` component:

- Component `Courses` should be opened by route `/courses`;
- If there is a token in the `localStorage`, then App navigates to the `/courses` by default.
- **You don't have to get courses and authors lists from backend.** Please, use mocked data as previous.
  We will use API for these data in the next module.
- When user clicks the `Add New Course` button the App navigates to `/courses/add`.

Add new course

Implement new features for `Courses` component:

- Component `CreateCourse` should be opened by router `/courses/add`.
- When user clicks on `Create course` button, App navigates to `/courses`
  (the new course should be in the course list in `Courses` component).

Header

Implement a new feature for `Header` component:

- Show user's name if he is logged in.
- When user clicks on `Logout` button, App should navigate to `/login`
  and you should remove `token` from localStorage.
- `Logout` button and user's name should not be on Login and Registration pages.

GOOD PRACTICES THAT YOU CAN APPLY FOR THIS TASK:

1. It's good practice to check response status, to provide default behavior for failed requests.