

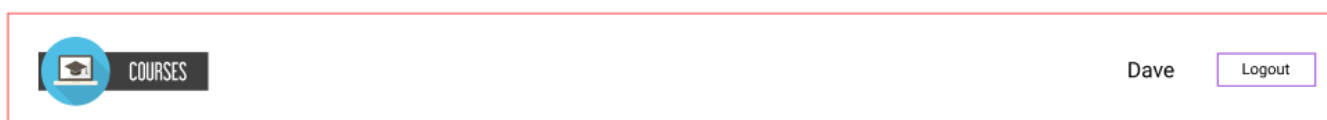
# Description

---

**Advise: just follow from one step to another.**

*The description was created in such a way that you do not need to navigate through the document at random.*

## Header



(example)

Create **Logo** component.

Logo component contains only logo picture (you can use any picture, you like).

PROMPT: **Logo component** is a just function component which returns **img** tag

Create **Button** component.

Button component is a function component which returns **button** tag.



### PAY ATTENTION

You will use **Button component** in several places through your app, so you should use **props** for this component such as **buttonText** and **onClick** (and any props, that you want).

Create **Header** component.

Header should be in **App** component.

**Header** component should include:

- **Logo** component;
- User's name (use any name for this module);
- **Button** component (**WITHOUT** any functionality for the current task).

---

## Courses

Create **CourseCard** component.

## Java

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 unchanged.

**Authors:** Dave Simmonds, Nikolas Le-Mark

**Duration:** 08:00 hours

**Created:** 01.02.2018

Show course

(example)

This component contains all info about course.

For the current task you should use `mockedCoursesList` array for courses info.

`CourseCard` components should be rendered in `Courses` component (you will create it in the next step).

`CourseCard` component should contain the following information:

1. Title (Course name);
2. Duration (format: hh:mm + 'hours');
3. Creation date;
4. Description;
5. `Show course` button (**WITHOUT** any functionality for the current task);
6. `Authors` list:
  - the authors' names should be displayed on the one line;
  - if all authors' names do not fit on one line, then the extra text should be cut off and '...' should be added at the end of line.



### PAY ATTENTION

In `mockedCoursesList` each course has only authors' `ids`.

To define authors' names you should find them in `mockedAuthorsList` by id.

Mocked courses and authors data. Save these data as constants and use in your App:

```
const mockedCoursesList = [
  {
    id: 'de5aaa59-90f5-4dbc-b8a9-aaf205c551ba',
    title: 'JavaScript',
    description: `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.` ,
        creationDate: '8/3/2021',
        duration: 160,
        authors: ['27cc3006-e93a-4748-8ca8-73d06aa93b6d', 'f762978b-61eb-4096-812b-
ebde22838167'],
    },
    {
        id: 'b5630fdd-7bf7-4d39-b75a-2b5906fd0916',
        title: 'Angular',
        description: `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.` ,
        creationDate: '10/11/2020',
        duration: 210,
        authors: ['df32994e-b23d-497c-9e4d-84e4dc02882f', '095a1817-d45b-4ed7-9cf7-
b2417bcbf748'],
    },
]

const mockedAuthorsList = [
    {
        id: '27cc3006-e93a-4748-8ca8-73d06aa93b6d',
        name: 'Vasiliy Dobkin'
    },
    {
        id: 'f762978b-61eb-4096-812b-ebde22838167',
        name: 'Nicolas Kim'
    },
    {
        id: 'df32994e-b23d-497c-9e4d-84e4dc02882f',
        name: 'Anna Sidorenko'
    },
    {
        id: '095a1817-d45b-4ed7-9cf7-b2417bcbf748',
        name: 'Valentina Larina'
    },
]

```

Create **Courses** component.

 COURSES

Vasya Logout

Search Add new course

### 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 unchanged.

**Authors:** Vasiliy Dobkin, Nicolas Kim

**Duration:** 02:40 hours

**Created:** 8.3.2021

Show course

### Angular

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.

**Authors:** Anna Sidorenko, Valentina Larina

**Duration:** 03:30 hours

**Created:** 10.11.2020

Show course

(example)

**Courses** component should be rendered in **App** component.

**Courses** component should include:

- **SearchBar** component (you will create it in the next step).
- **CourseCard** component.
- **Add new course** button. (Reuse **Button** component)

#### PROMPT:

You should import **mockedCoursesList** and **mockedAuthorsList** to **Courses** component.

Use loop for **CourseCard** rendering.

Please, follow [this link](#) and scroll down to the **"Embedding map() in JSX"** topic to see the example.

Search



(example)

Create **Input** component.

**Input component** should contain **input** and **label** tags.

Using a **label** tag for **input** is considered good practice. [Usage and benefits of label tag.](#)



#### PAY ATTENTION

You will use **Input component** in several places through your app, so you should use **props** for this component such as **labelText**, **placeholderText** and **onChange** (and any props, that you want).

Create **SearchBar** component.

**SearchBar** should be in Courses component (above the list of courses).


**SearchBar** component should include:

- **Input** component;
- **Button** component.

**Add functionality for searching courses:**

- User should have ability to search course by **title** and **id**;
- The search is performed by the occurrence of characters in the string, and **not just by a match at the beginning of the string**;
- **Case-insensitive** search;
- When user clicks on **Search** button it displays all courses that match the search query;
- All courses are displayed when user cleans search field.

(Search by title example)

 COURSES

Vasya [Logout](#)

[Search](#) [Add new course](#)

## 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 unchanged.

**Authors:** Vasiliy Dobkin, Nicolas Kim  
**Duration:** 02:40 hours  
**Created:** 8.3.2021

[Show course](#)

---

Add new course

**COURSES** Vasya Logout

Title  
Enter title... Create course

Description  
Enter description

**Add author**  
Author name  
Enter author name... Create author

**Authors**  
Vasiliy Dobkin Add author  
Nicolas Kim Add author  
Anna Sidorenko Add author  
Valentina Larina Add author

**Duration**  
Duration  
Enter duration in minutes...  
Duration: 00:00 hours

**Course authors**  
Author list is empty

(example)

Create **CreateCourse** component.

This component should be rendered instead of the **Courses** component if user clicks **Add New Course** button.

**CreateCourse** component should contain the following elements:

- **Title** (input) - field for input course name;
- **Description** (textarea) - text length should be at least 2 characters;
- **Authors** - contains a list of all authors and their corresponding **Add author** buttons;
- **Course authors** - contains a list of authors course and their corresponding **Delete author** buttons;
- **Delete author** - when user clicks on this button the corresponding author disappears from the **Course authors** list and shows in **Authors**;
- **Add author** - when user clicks on this button the corresponding author disappears from the **Authors** list and shows in **Course authors**;

- **Author field** (input) - author name length should be at least 2 characters;
- **Create author** (button) - when user clicks on this button:
  - the new author appears in **Authors**;
  - the new author adds to array with all authors;
  - the author's id generates automatically;
  - new author info should be presented as an object (see model of Author below);
  - for the current task new author should be added to `mockedAuthorsList`.

Course model:

```
// Course
{
  id: string
  title: string
  description: string
  creationDate: string
  duration: number
  authors: [authorId]
}
```

Author model:

```
// Author
{
  id: string
  name: string
}
```

- **Duration** - this part provides logic for adding course duration time.
  - the duration of the course is entered in minutes;
  - for the correct display of the course duration, you need to format minutes into hours and minutes;
  - duration should be more than 0 minute;
  - user should have an ability to enter ONLY numbers into the field.
- **Create course** (button) - when user clicks on this button:
  - `CreateCourse` component closes;
  - `Courses` component displays;
  - list of courses displays with the new course.

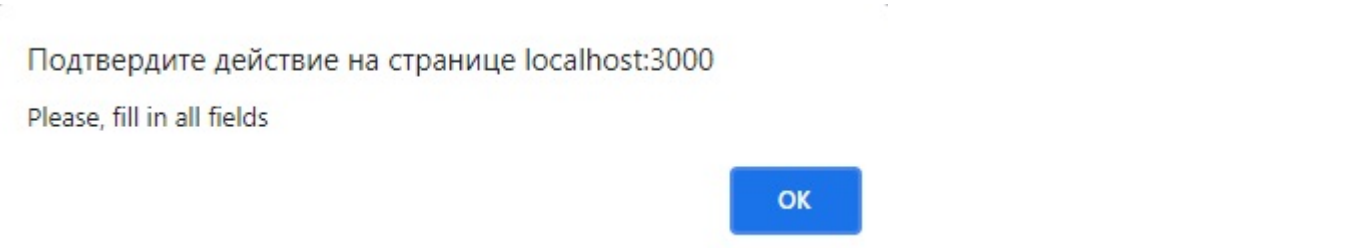
`Create course` button should always be available.




ALL FIELDS ARE REQUIRED



In case, when the user clicks on the **Create course** button, and some field is not filled in, then an **alert** message should be displayed



(alert message example)

 COURSES


Vasya Logout

Title

Create course

Description

test description text



Add author

Author name

Create author

Duration

Duration: 02:03 hours

Authors

Nicolas Kim

Anna Sidorenko

Valentina Larina

Add author

Add author

Add author

Course authors

Vasiliy Dobkin

Delete author

(example of filled **CreateCourse** component)

- PROMPT: Study [this documentation](#) to do **conditional rendering** for **CreateCourse** component
- Field **id** for course and for the author should be generated automatically.
- To do this, you can use the [UUID library](#) or any way you like.
- Field **creationDate** generated automatically base on current date. Date format: d/m/yyyy (see examples in mockedCoursesList)

---

# GOOD PRACTICES THAT YOU CAN APPLY FOR THIS TASK:

---

1. Place `css` files in the folder with the component.

Example:

```
src
|
|- components
|   |
|   |-Header
|       |-Header.jsx //component
|       |-header.css //styles for this component
```

2. Use `label` tag for input. [Link to the W3C](#).
3. Use `key` attribute for list rendering. [Link to the documentation](#).
4. Use constants to avoid a hardcode. *For example, when passing string properties to a component:*

```
// BAD
<Button text='Search' />

// GOOD
//constants.js
const BUTTON_TEXT = 'Search';

//Button.jsx
import {BUTTON_TEXT} from './constants.js';

<Button text={BUTTON_TEXT} />
```

5. Add empty string between import groups. *Example:*

```
import from '[node_modules]'

import from '[own_components]'
import from '[constants]'

import from '[styles]'
```

6. Use common `index.js` file for components and helpers. [Link to the article](#)

7. If component does not have any logic you can return the markup immediately. *Examples:*

```
import React from 'react';

import logo from '../assets/logo.jpg';

export const Logo = () => <img src={logo} />;
```

or

```
import React from 'react';

export const Button = ({ text, onClick }) => (
  <button onClick={onClick}>{text}</button>
);
```