

# Список дел

## Цель работы

Познакомить и получить навыки в реализации списка дел с помощью React-компонентов.

## Задания для выполнения

1. Реализуйте веб-приложение «Список дел» с помощью стандартного React-компонента:

```
class TodoApp extends React.Component {  
  constructor(props) {  
    super(props);  
    this.state = { items: [], text: '' };  
    this.handleChange = this.handleChange.bind(this);  
    this.handleSubmit = this.handleSubmit.bind(this);  
  }
```

```
  render() {  
    return (  
      <div>  
        <h3>TODO</h3>  
        <TodoList items={this.state.items} />  
        <form onSubmit={this.handleSubmit}>  
          <label htmlFor="new-todo">  
            What needs to be done?  
          </label>  
          <input  
            id="new-todo"  
            onChange={this.handleChange}  
            value={this.state.text}  
          />  
          <button>  
            Add #{this.state.items.length + 1}  
          </button>  
        </form>  
      </div>  
    );  
  }
```

```
  handleChange(e) {  
    this.setState({ text: e.target.value });  
  }
```

```
  handleSubmit(e) {  
    e.preventDefault();
```

```

    if (this.state.text.length === 0) {
      return;
    }
    const newItem = {
      text: this.state.text,
      id: Date.now()
    };
    this.setState(state => ({
      items: state.items.concat(newItem),
      text: ''
    }));
  }
}

class TodoList extends React.Component {
  render() {
    return (
      <ul>
        {this.props.items.map(item => (
          <li key={item.id}>{item.text}</li>
        ))}
      </ul>
    );
  }
}

ReactDOM.render(
  <TodoApp />,
  document.getElementById('todos-example')
);

```

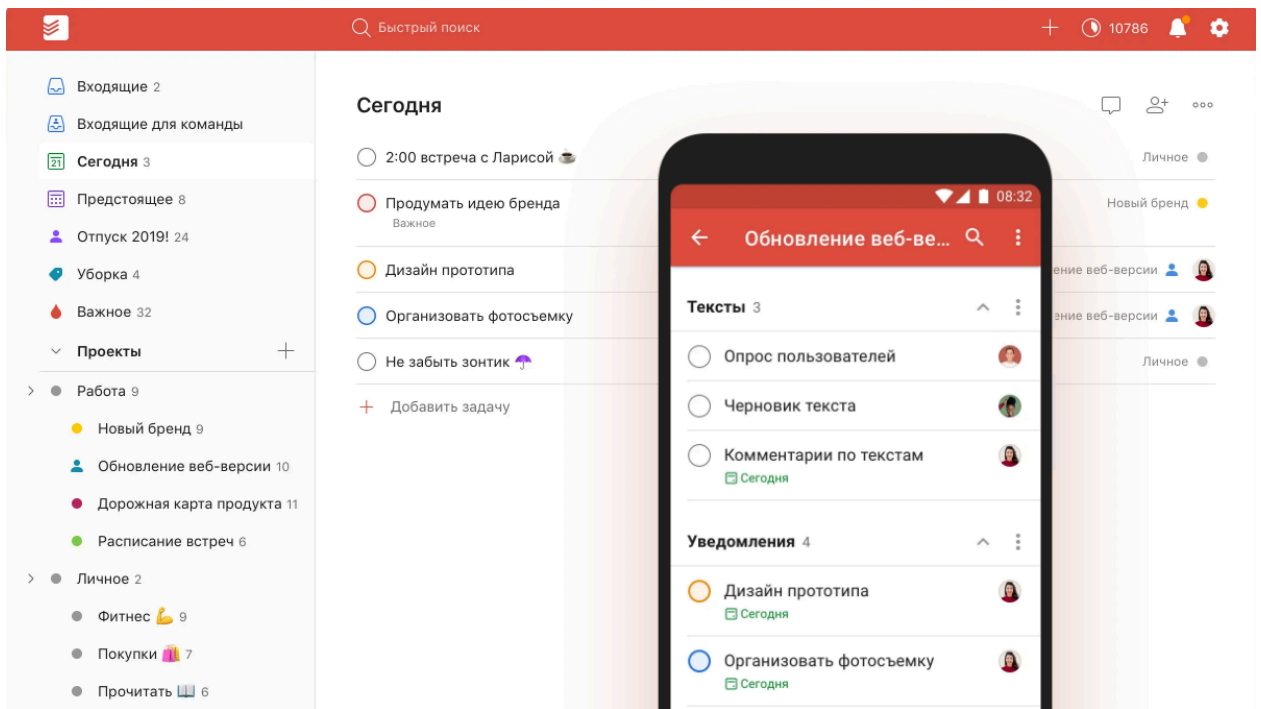
## Техническое задание

Приложение должно содержать меню (более 4х элементов) – группировка дел по категориям (личные, работа, учеба, отдых и т.д.) с возможностью вести список в каждом из категорий.

Оформление страниц не должно включать шрифты с засечками и нечитабельные цветовые сочетания.

Каждая категория должна сопровождаться пиктограммой.

Пример для реализации:



2. Загрузить созданную страницу на GitHub в репозиторий tera, используя формат в названии Фамилия (латинскими буквами)\_13.

## Методические указания

Для реализации вашего проекта можно использовать любую структуру проекта. Например, реализацию внутри html-страницы:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <meta charset="utf-8" />
```

```
  <title>Hello React</title>
```

```
</head>
```

```
<body>
```

```
  <div id="app"></div>
```

```
  <script crossorigin src="https://unpkg.com/react@16/umd/react.production.min.js"></script>
```

```
  <script crossorigin  
src="https://unpkg.com/react-dom@16/umd/react-dom.production.min.js"></script>
```

```
  <script src="https://cdnjs.cloudflare.com/ajax/libs/babel-standalone/6.25.0/babel.min.js"></script>
```

```
  <script type="text/babel">
```

```
  ....
```

Бутстрап можно подключить по инструкции:

<https://itchief.ru/bootstrap/installation>

Сочетание читабельных цветосочетаний можно проверить здесь:

<https://colorable.jxnblk.com/fefff/9cc2a5>

Код пиктограмм можно взять здесь:

<http://konstantinbulgakov.com/tools/icons>

### **Дополнительные задания:**

1. Добавьте другие компоненты React (слайдеры, виджеты, карточки дел, анимацию и т.д.), создайте для них функционал.