

МГТУ им. Н. Э. Баумана
Кафедра «Системы обработки информации и управления»

Лабораторная работа №8
по курсу «Разработка интернет-приложений»
«Разработка пользовательского интерфейса с использованием библиотеки
React»

Выполнила:
Ларионова А.П., ИУ5-53Б
Преподаватель:
Гапанюк Ю.Е.

Москва, 2021 год

Задание:

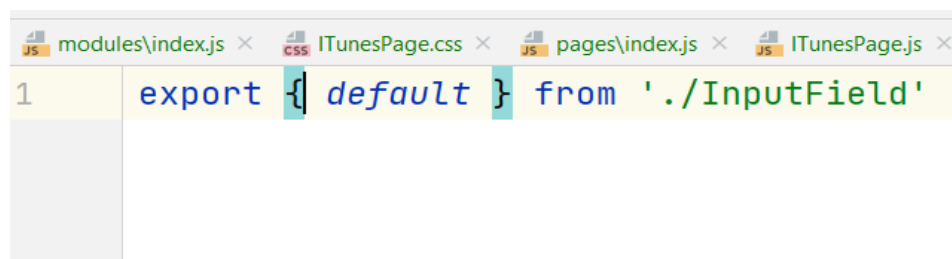
На основе [методических указаний](#) разработайте React-приложение. Для создания приложения необходимо решить следующие задачи:

1. Создать стартовый React-проект. Удалить неиспользуемый код. Организовать директории для страниц, компонентов, утилит и работы с сетью.
2. Организовать роутинг в веб-приложении.
3. Разработать базовые страницы, на которых будут отображаться сущности из выбранной вами предметной области:
 - i. Стартовая страница.
 - ii. Страница просмотра списка объектов.
 - iii. Страница просмотра конкретного объекта.
4. Вынести переиспользуемые компоненты в отдельные файлы:
 - i. Для навигации по приложению можно добавить header.
 - ii. Для отображения дополнительной информации (данные о студенте и предметной области) можно использовать footer.
 - iii. Источники ввода-вывода (поля ввода (inputs)/формы/текстовые блоки).
 - iv. Переиспользуемые таблицы/гриды.
5. Добавить асинхронные запросы в разработанный API, чтобы страница получала данные с сервера.
6. Если в Вашем проекте реализована сложная логика работы с состоянием приложения, то рекомендуется добавить пользовательские хуки.
7. Страницы приложения должны хорошо отображаться как на больших, так и на маленьких экранах.

Текст программы:

Src/components/TextField

Index.js



```
1 export { default } from './TextField'
```

TextField.css

```
is  _IT_ window Help  ojangoproject8lr - InputField.css

css InputField.css

s\index.js x CSS iTunesPage.css x JS pages\index.js x JS iTunesPage.js x CSS MusicCard.css x JS MusicCard\index.js x CSS InputField.css x

.inputField{
  display: flex;
  padding: 12px 0;
}

.inputField > input{
  width: 200px;
  margin-right: 12px;
  border: 2px solid black;
  border-radius: 4px;
  outline: none;
}
```

InputField.js

```
modules\index.js x CSS iTunesPage.css x JS pages\index.js x JS iTunesPage.js x CSS MusicCard.css x JS MusicCard\index.js x CSS InputField.css x JS InputField.js x JS InputField\index.js x JS Database x JS SolView

1 import {Button} from "react-bootstrap";
2 import React from "react";
3 import './InputField.css';
4
5 const InputField = ({ value, setValue, onSubmit, loading, placeholder, buttonText : string = 'Поиск'}) =>
6   return <div className="inputField">
7     <input value={value} placeholder={placeholder} onChange={(event => setValue(event.target.value))
8     <Button disabled={loading} onClick={onSubmit}>{buttonText}</Button>
9   </div>
10
11
12 export default InputField
```

Src/components/MusicCard

Index.js

```
modules\index.js x CSS iTunesPage.css x JS pages\index.js x JS iTunesPage.js x CSS MusicCard.css x JS MusicCard\index.js x

1 export { default } from './MusicCard';
```

MusicCard.css

```
modules\index.js x iTunesPage.css x pages\index.js x iTunesPage.js x MusicCard.css x
1  .card{
2    padding: 8px;
3  }
4
5  .loadingBg{
6    z-index: 999;
7    position: fixed;
8    height: 100%;
9    width: 100%;
10   display: flex;
11   justify-content: center;
12   align-items: center;
13 }
14
15 .textStyle{
16   height: 48px;
17   overflow: hidden;
18   text-overflow: ellipsis;
19   margin-bottom: 12px;
20 }
21 .cardImage{
```

MusicCard.js

```
modules\index.js x iTunesPage.css x pages\index.js x iTunesPage.js x MusicCard.css x MusicCard\index.js x InputField.css x InputField.js x InputField\index.js x MusicCard.js x
1  import {Button, Card} from "react-bootstrap";
2  import React from "react";
3  import './MusicCard.css';
4
5  const MusicCard = ({artworkUrl100, artistName, collectionCensoredName, trackViewUrl}) => {
6
7    return <Card className="card">
8      <Card.Img className="cardImage" variant="top" src={artworkUrl100} height={100} width={100} />
9      <Card.Body>
10        <div className="textStyle">
11          <Card.Title>{artistName}</Card.Title>
12        </div>
13        <div className="textStyle">
14          <Card.Text>
15            {collectionCensoredName}
16          </Card.Text>
17        </div>
18        <Button className="cardButton" href={trackViewUrl} target="_blank" variant="primary">Открыть в iTunes</Button>
19      </Card.Body>
20    </Card>
21  }
```

Src/hooks

```

1 import {useEffect, useState} from "react";
2
3 // Пример пользовательского хука (Называйте пользовательские хуки с use в начале!)
4
5 export const useWindowSize = () => {
6     // в данном пользовательском хуке мы используем хук состояния и хук эффекта
7     const [windowSize, setWindowSize] = useState( initialState: {
8         width: undefined,
9         height: undefined,
10    });
11    useEffect( effect: () => {
12        // при вызове этой функции, мы будем "класть" в состояние актуальную высоту и ширину экрана
13        function handleResize() {
14            setWindowSize( value: {
15                width: window.innerWidth,
16                height: window.innerHeight,
17            });
18        }
19        // В данном примере мы будем подписываться на изменение размеров экрана, чтобы всегда иметь актуаль
20        window.addEventListener( type: "resize", handleResize);
21        handleResize();
22    });
23
24    // В данном примере мы будем подписываться на изменение размеров экрана, чтобы всегда иметь актуаль
25    window.addEventListener( type: "resize", handleResize);
26    handleResize();
27    // После того, как компонент "уничтожается", желательно избавиться от всех "слушателей", чтобы не т
28    return () => window.removeEventListener( type: "resize", handleResize);
29    }, deps: []);
30
31    return windowSize;
32 }

```

Src/modules/index.js

```

1 const DEFAULT_SEARCH_VALUE = 'radiohead';
2
3 export const getMusicByName = async (name : string = DEFAULT_SEARCH_VALUE) =>{
4     const res = await fetch( input: `https://itunes.apple.com/search?term=${name}` )
5     .then((response : Response ) => {
6         return response.json();
7     }).catch(()=>{
8         return {resultCount:0, results:[]}
9     })
10    return res
11 }
12

```

Src/pages/index.js

```
modules\index.js x pages\index.js x ITunesPage.js x MusicCard.css x
1 export { default } from './StartPage'
2 |
```

ITunesPage.css

```
modules\index.js x pages\index.js x ITunesPage.css x MusicCard.css x
1 .container{
2     display: flex;
3     flex-direction: column;
4     padding: 20px 40px;
5     min-width: 800px;
6 }
7 .containerLoading{
8     filter: blur(8px);
9 }
10
11 @media (max-width: 600px) {
12     .container{
13         min-width: auto;
14     }
15 }
```

ITunesPage.js

```
modules\index.js x pages\index.js x ITunesPage.css x ITunesPage.js x MusicCard.css x MusicCard\index.js x InputField.css x InputField.js x MusicCard.js x
1 import React, {useState} from 'react';
2 import { Col, Row, Spinner } from "react-bootstrap";
3 import MusicCard from "../components/MusicCard";
4 import { getMusicByName } from '../modules';
5 import './ITunesPage.css';
6 import {useWindowSize} from "../hooks/useWindowSize";
7 import InputField from "../components/InputField";
8 function ITunesPage() {
9
10     const [searchValue, setSearchValue] = useState( initialState: 'radiohead');
11
12     const [filter, setFilter] = useState( initialState: '');
13
14     const [loading, setLoading] = useState( initialState: false)
15
16     const [music, setMusic] = useState( initialState: [])
17
18     const handleSearch = async () => {
19         // Сбрасываем фильтр
20         await setFilter( value: '');
21         // Ставим задержку
```

```

19 // Сбрасываем фильтр
20 await setFilter( value: '');
21 // Ставим загрузку
22 await setLoading( value: true);
23 const { results } = await getMusicByName(searchValue);
24 // Добавляем в состояние только треки
25 await setMusic(results.filter(item => item.wrapperType === "track"));
26 // Убираем загрузку
27 await setLoading( value: false)
28 }
29
30 const handleFilter = ()=> {
31   setMusic( value: music => music.filter(item=>item.artistName && item.artistName.includes(filter)));
32 }
33
34 const { width } = useWindowSize();
35 const isMobile = width && width <= 600;
36
37 return (
38   <div className={`container ${loading && 'containerLoading'}`}>
39     {loading && <div className="loadingBg"><Spinner animation="border"/></div>}

```

```

34 const { width } = useWindowSize();
35 const isMobile = width && width <= 600;
36
37 return (
38   <div className={`container ${loading && 'containerLoading'}`}>
39     {loading && <div className="loadingBg"><Spinner animation="border"/></div>}
40     <InputField value={searchValue} setValue={setSearchValue} placeholder="поиск" loading={loading}
41     <InputField value={filter} setValue={setFilter} placeholder="Автор" loading={loading} onSubmit=
42     {!music.length ? <h1>К сожалению, пока ничего не найдено :(</h1>:
43       <Row xs={1} md={isMobile ? 1 : 4} className="g-4">
44         {music.map((item, index : number )=>{
45           return isMobile ? <MusicCard {...item} key={index}/> :(
46             <Col key={index}>
47               <MusicCard {...item}/>
48             </Col>
49         )
50       )}}
51     </Row>
52   }
53 </div>
54 );

```

```

50   )}}
51 </Row>
52 }
53 </div>
54 );
55 }
56
57 export default ITunesPage;

```

StartPage.css

```
modules\index.js × pages\index.js × ITunesPage.css × ITunesPage.js × StartPage.css ×
1  .buttons{
2      display: flex;
3      flex-wrap: wrap;
4      align-items: baseline;
5  }
6  .button{
7      width: 128px;
8      margin: 16px;
9  }
10
11  .button-with-title{
12      display: flex;
13      flex-direction: column;
14  }
15  .button-with-title > button{
16      width: 128px;
17  }
```

StartPage.js

```
modules\index.js × pages\index.js × ITunesPage.css × ITunesPage.js × StartPage.css × StartPage.js × MusicCard.css × MusicCard\index.js × MusicCard.js × useWindowSize.js ×
1  import React, {useEffect, useState} from 'react';
2  import './StartPage.css';
3
4  const data = [
5      'Берик Дондаррион',
6      'Леди Мелиссандра',
7      'Полливер',
8      'Уолдер Фрей',
9      'Тайвин Ланнистер',
10     'Сир Мерин Трэнт',
11     'Король Джоффри',
12     'Сир Илин Пейн',
13     'Гора',
14     'Пес',
15     'Серсея Ланнистер',
16 ]
17
18 function StartPage() {
19
20     // В функциональных компонентах для работы с состоянием можно использовать хук useState()
21     // Он возвращает кортеж из двух элементов:
```



```

19
20 // В функциональных компонентах для работы с состоянием можно использовать хук useState()
21 // Он возвращает кортеж из двух элементов:
22 // 1 элемент - объект состояния
23 // 2 элемент - метод который позволит нам обновить состояние
24 const [randomName, setRandomName] = useState();
25
26 // Кстати, это хороший пример деструктуризации массива в JavaScript
27 const [names, setNames] = useState(data);
28
29 const [showNames, setShowNames] = useState(initialState: false);
30
31 // В данном хендлере мы изменяем состояние на какое-то конкретное
32 const handleShowNames = () =>{
33     setShowNames( value: true)
34 }
35
36 // В данном хендлере мы изменяем состояние на какое-то конкретное
37 const handleHideNames = () =>{
38     setShowNames( value: false)
39 }

```

```

37 const handleHideNames = () =>{
38     setShowNames( value: false)
39 }
40
41 // В данном хендлере мы изменяем состояние в зависимости от его прошлого значения
42 const handleSwitch = () =>{
43     // метод из useState может принимать как определенное значение, так и метод,
44     // принимающий прошлое значение и возвращающий новое
45     setShowNames( value: state => !state)
46 }
47
48 useEffect( effect: ()=>{
49     console.log('Этот код выполняется только на первом рендере компонента')
50     // В данном примере можно наблюдать Spread syntax (Трехточие перед массивом)
51     setNames( value: names=> [...names, 'Бедный студент'])
52
53     return () => {
54         console.log('Этот код выполняется, когда компонент будет размонтирован')
55     }
56 }, deps: [])
57

```

```

55     }
56   }, deps: []])
57
58   useEffect( effect: ()=>{
59     console.log('Этот код выполняется каждый раз, когда изменится состояние showNames ')
60     setRandomName(names[Math.floor( x: Math.random()*names.length)])
61   }, deps: [showNames])
62
63
64   return (
65     <div>
66       <h3>Случайное имя из списка: {randomName}</h3>
67       <div className="buttons">
68         {/*Кнопка для того, чтобы показать имена*/}
69         <button className="button" onClick={handleShowNames}>Хочу увидеть список имен</button>
70         {/*Кнопка для того, чтобы скрыть имена*/}
71         <button className="button" onClick={handleHideNames}>Хочу скрыть список имен</button>
72
73         {/*Универсальная кнопка*/}
74         <div className="button-with-title">
75           <button onClick={handleSwitch}>{showNames ? 'Хочу скрыть список имен' : 'Хочу увидеть список

```

```

73     {/*Универсальная кнопка*/}
74     <div className="button-with-title">
75       <button onClick={handleSwitch}>{showNames ? 'Хочу скрыть список имен' : 'Хочу увидеть список
76       <span>Эта кнопка переключает отображение списка</span>
77     </div>
78   </div>
79   {/*React отрисует список только если showNames будет равен true, boolean значения игнорируются
80   {showNames && <ul>
81     {/*Рекомендую почитать про прекрасные встроенные методы массивов в JavaScript */}
82     {names.map((name : string , index : number )=>{
83       return (
84         <li key={index}>
85           <span>{name}</span>
86         </li>
87       )
88     }}}
89   </ul>
90   }
91   </div>
92 );
93 }

```

```

87   )
88   }}}}
89   </ul>
90   }
91   </div>
92 );
93 }
94
95 export default StartPage;|
96

```

App.js

```

1 import { BrowserRouter, Route, Link, Switch } from "react-router-dom";
2 import StartPage from "../pages/StartPage.js";
3 import ITunesPage from "../pages/ITunesPage.js";
4 import { Navbar, Container, Nav, NavDropdown } from 'react-bootstrap'
5 function App() {
6
7     return (
8         <BrowserRouter basename="/" >
9             <Navbar bg="light" expand="lg">
10                 <Container>
11                     <Navbar.Brand href="#home">Лаба 8</Navbar.Brand>
12                     <Navbar.Toggle aria-controls="basic-navbar-nav" />
13                     <Navbar.Collapse id="basic-navbar-nav">
14                         <Nav className="me-auto">
15                             <Link to="/"><span style={{margin:'0 12px'}}>Стартовая страница</span></Link>
16                             <br/>
17                             <Link to="/new"><span style={{margin:'0 12px'}}>Хочу на страницу с чем-то новеньким(Пустая
18                             <br/>
19                             <Link to="/music"><span style={{margin:'0 12px'}}>Хочу на страницу с рабочим апи ITunes</span></Link>
20                         </Nav>

```

```

19         <Link to="/music"><span style={{margin:'0 12px'}}>Хочу на страницу с рабочим апи ITu
20         </Nav>
21     </Navbar.Collapse>
22 </Container>
23 </Navbar>
24 <div>
25     <Switch>
26         <Route exact path="/">
27             <StartPage/>
28         </Route>
29         <Route path="/new">
30             <h1>Это наша страница с чем-то новеньким</h1>
31         </Route>
32         <Route path="/music">
33             <ITunesPage/>
34         </Route>
35     </Switch>
36 </div>
37 </BrowserRouter>
38 );
39 }

```

```

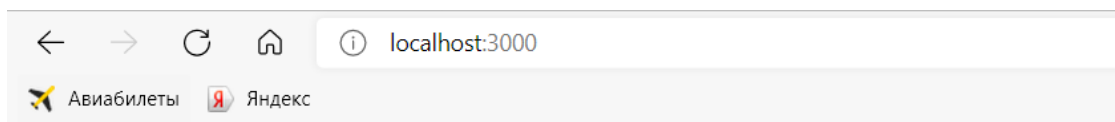
32         <Route path="/music">
33             <ITunesPage/>
34         </Route>
35     </Switch>
36 </div>
37 </BrowserRouter>
38 );
39 }
40
41 export default App;

```

Index.js

```
pages\index.js x iTunesPage.css x iTunesPage.js x StartPage.css x StartPage.js x App.css x App.js x App.test.js x index.css x src\index.js x
1 import React from 'react';
2 import ReactDOM from 'react-dom';
3 import './index.css';
4 import App from './App';
5 import reportWebVitals from './reportWebVitals';
6
7 ReactDOM.render(
8   <React.StrictMode>
9     <App />
10  </React.StrictMode>,
11  document.getElementById( elementId: 'root')
12 );
13
14 // If you want to start measuring performance in your app, pass a function
15 // to log results (for example: reportWebVitals(console.log))
16 // or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
17 reportWebVitals();
```

Экранные формы:



Лаба 8

[Стартовая страница](#)

[Хочу на страницу с чем-то новеньким\(Пустая страничка\)](#)

[Хочу на страницу с рабочим апи iTunes](#)

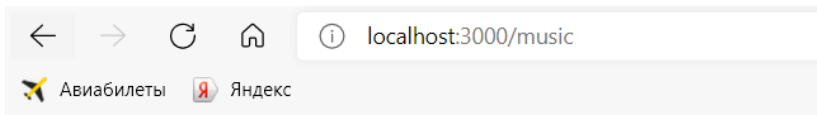
Случайное имя из списка: Полливер

Хочу увидеть
список имен

Хочу скрыть
список имен

Хочу увидеть
список имен

Эта кнопка переключает отображение списка



Лаба 8

[Стартовая страница](#)

[Хочу на страницу с чем-то новеньким\(Пустая страничка\)](#)

[Хочу на страницу с рабочим api iTunes](#)

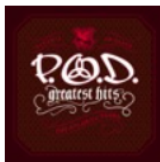
SNOOP DOGG



Snoop Dogg

Doggumentary (Bonus Track Version)

[Открыть в iTunes](#)



P.O.D.

Greatest Hits: The Atlantic Years

Лаба 8

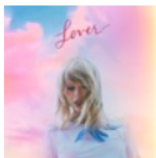
[Стартовая страница](#)

[Хочу на страницу с чем-то новеньким\(Пустая страничка\)](#)

[Хочу на страницу с рабочим апи iTunes](#)

Искать

Отфильтровать



Taylor Swift

Lover

[Открыть в iTunes](#)



Taylor Swift

Lover

← → ↺ 🏠 📄 ⚙️ ☆ 📁 👤 ...
🛩️ Авиабилеты 🇾🇵 Яндекс

Apple Music

🔍 Search

🎧 Listen Now

📀 Browse

📻 Radio

📱 Open in iTunes ➔

Lover
Taylor Swift
POP · 2019

There's a reason Taylor Swift sounds so confident and cool on *Lover*, her seventh album and the most free-spirited yet. She's in love—pure, steady, starry-eyed, shout-it-from-the-rooftops love. Arriving 13 **MORE**

▶ Preview

1	I Forgot That You Existed	2:50	...
★ 2	Cruel Summer	2:58	...
★ 3	Lover	3:41	...

Sign In