|  |
| --- |
| https://lh6.googleusercontent.com/QcftzNtI05T0Y6fjdSh1Rr2rt8oqZ1IvnLvbn1jLJ7CCyteVir3k-xBLv4SL1wAgWJsRhmmJSR0UW-RP63_GQenE4vVWv05BRoZTsmIcBccVTnfxwmsnNMvjg599x9SqZd8E3dkd |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждение высшего образования«МИРЭА - Российский технологический университет»РТУ МИРЭА |

Институт информационных технологий (ИТ)

Кафедра инструментального и прикладного программного обеспечения (ИиППО)

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ №5** | |
| **по дисциплине** | |
| «Разработка клиент-серверных приложений»  на тему  **«Redux, события и бизнес логика.»** | |
| Выполнил студент группы ИКБО-20-19 | Анваржонов Ж Т |
|  |  |
| Принял | ассистент, Коваленко М.А. |

Практическая работа выполнена «10» февраля 2022 г.

(подпись студента)

Зачтено «\_\_» 2022 г.

(подпись студента)

Москва 2022

**Цель работы:** создать простое React приложение.

**Задание:**

1. Cоздать новый проект
2. Создать форму принятия пользовательского соглашения
   1. Форма должна содержать чекбокс, который следует отметить пользователем, в случае, если он принимает соглашение
   2. Форма должна содержать кнопку подтверждения соглашения, которая становится активной только при активации чекбокса

**Описание выполнения работы:**

Листинг 1 – листинг Action.ts:

*import* { CHECKBOX\_CLICK } *from* 'actionTypes'  
  
*export const* checkboxClick = (name: *string*) => ({ type: CHECKBOX\_CLICK, payload: name })

Напишем обработчик события изменения состояния чекбокса, т. Е. Reducer

Листинг 2 – листинг reducers.ts:

*import* { CHECKBOX\_CLICK } *from* 'actionTypes'  
  
*const* initialState: Record<*string*, *boolean*> = {}  
  
*export const* checkboxReducer = (  
 state = initialState,  
 action: { type: *string*; payload?: *string* },  
) => {  
 *if* (action.type === CHECKBOX\_CLICK && action.payload) {  
 *return* { ...state, [action.payload]: !state[action.payload] }  
 }  
 *return* state  
}

Чтобы получить состояние чекбокса, напишем selector.

Листинг 3 – структура хранения для селектора

*export type* State = {  
 checkboxes: Record<*string*, *boolean*>  
}

Листинг 3 – листинг кода selectors.ts

*import* { State } *from* 'types'  
  
*export const* getCheckboxState = (name: *string*) => (state: State) => state.checkboxes[name] ?? *false*

Cоздадим сам Store, т.е. специальный объект Redux, в котором хранится состояние приложения.

Листинг 4 – листинг кода store.ts

*import* { AnyAction, createStore, combineReducers } *from* 'redux'  
*import* { State } *from* 'types'  
*import* { checkboxReducer } *from* 'reducers'  
  
*export const* createAppStore = (initialState?: State) =>  
 createStore<State, AnyAction, *unknown*, *unknown*>(  
 combineReducers({  
 checkboxes: checkboxReducer,  
 }),  
 initialState,  
 )  
  
*export default* createAppStore()

Создадим checkbox:

Листинг 5- листинг кода checkbox

*import* React *from* 'react'  
*import* { useDispatch, useSelector } *from* 'react-redux'  
*import* { getCheckboxState } *from* 'selectors'  
*import* { checkboxClick } *from* 'actions'  
  
*const* Checkbox: React.FC<{ name: *string*; label: *string* }> = ({ name, label }) => {  
 *const* dispatch = useDispatch()  
 *const* checked = useSelector(getCheckboxState(name))  
  
 *const* handleClick = React.useCallback(() => {  
 dispatch(checkboxClick(name))  
 }, [dispatch, name])  
  
 *return* (  
 <div>  
 <input id={name} type="checkbox" checked={checked} onChange={handleClick} />  
 <label htmlFor={name}>{label}</label>  
 </div>  
 )  
}  
  
*export default* Checkbox

Подключим redux store в само приложение

*import* React *from* 'react'  
*import* ReactDOM *from* 'react-dom'  
*import* { Provider } *from* 'react-redux'  
*import* 'index.css'  
*import* store *from* 'store'  
*import* App *from* 'App'  
  
ReactDOM.render(  
 <React.StrictMode>  
 <Provider store={store}>  
 <App />  
 </Provider>  
 </React.StrictMode>,  
 document.getElementById('root'),  
)

Покажем как выглядит наше приложение:

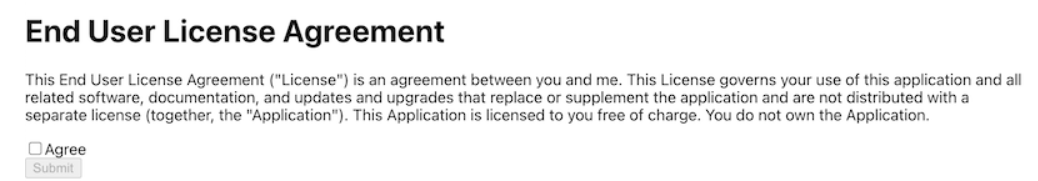


Рисунок 1 – внешний вид страницы

**Вывод:** В данной практической работе был освоены и применены навыки реализации формы принятия пользовательского соглашения с помощью redux. В ходе выполнения работы улучшены навыки разработки React-приложения.