1) Поставити 2 пакета react-redux redux

2) Створюємо в папці src папку redux

3) в папці redux створюємо 3 папки store, actions, reducers

4) в index.js обгортаємо наш App в компонент Provider і передаємо йому пропсом store силку на обєкт storе, що буде створений в папці redux/store

5) Створюємо файл store.js який знаходиться за адресою redux/store/store.js;

6) імпортуємо в файл store.js метод createStore з бібліотеки redux

7) підключаємо REDUX DEVTOOLS до проекта створюючи константу const DevTools = window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_ && window.\_\_REDUX\_DEVTOOLS\_EXTENSION\_\_();

8) Створюємо 2 файла в папці redux/reducers

8.1) index.js - всі редюсери обєднуються в один обєкт

8.2) counerReducer.js - файл в якому створиться поле для глобального стора.

9) створюємо ф-ю reducer в якій через switch описуємо як буде мінятися стейт в залежності від типу екшена що запускається(від команди);

10) import { combineReducers } from 'redux' в файлі index.js що знаходиться в папці redux/reducers;

11) імпортуємо створений редюсер в файл index.js що знаходиться в папці redux/reducers;

12) створюємо змінну rootReducer куди записуємо виклик метода combineReducers для того щоб сформувати фінальний обєкт стейта всієї програми. (записуємо в обєкт ключі з імпортів що робимо в пункті №11); Не забути додати export default rootReducer для того щоб можна було цю змінну використати в пункті №13.

13) в store.js імпортуємо змінну rootReducer яка знаходиться за адресою redux/reducers/index.js;

14) в файлі store.js створюємо змінну store куди записуємо виклик метода createStore(). Аргументами передаємо змінну rootReducer з пункта №13 і змінну DevTools з пункта № 7; І експортуємо змінну store для подальшого використання.

15) в файл index.js імпортуємо константу store з файла store.js і передаємо її пропсом в компонент Provider для того щоб підключити глобальний обєкт store до нашого App.

16) Створюємо екшени - ф-ї що повертають обєкт з полем type в якому написана команда на яку має реагувати редюсер.

Команда в case редюсера має співпадати з командою в полі type для екшена.

17) Вся взаємодія між користувачем і програмою відбувається через екшени. Тому якщо ми хочемо міняти стейт через кліки чи інші події, то потрібно імпортувати екшени в компонент де буде відбуваєтися навішування на кнопки.

18) Якщо ми хочемо підєднати компонент до store нам потрібно імпортнути в компонент метод connect з пакета react-redux.

19) Створюємо ф-ю mapStateToProps яка приймає параметром store і повертає обєкт. В обєкті ми пишемо назву пропса за якою будемо звертатися до поля зі store в нашому компоненті.

20) Створюємо ф-ю mapDispatchToProps яка приймає параметром метод dispatch який буде активувати наші екшени. Ця ф-я також повертає обєкт з полями в яких записані ф-ї що викликають метод dispatch() аргументами якого є екшени що імпортовані в компонент в пункті № 17.

21) Щоб підєднати компонент до стора використовуємо метод connect що імпортували в компонент в пункті № 18. Цей метод приймає аргументами 2 ф-ї. Перший аргумент це ф-я mapStateToProps з пункта № 19 яка дає доступ компоненту до конкретного поля в глобальному стейті. Другий аргумент це ф-я mapDispatchToProps з пункта № 20, що дає доступ до активації екшенів через метод dispatch. Записується це так

connect(mapStateToProps,mapDispatchToProps)(назва компонента);

22) Після connect ми маєм доступ до полів стора і екшенів через пропси. В тупому компоненті через props. В розумному через this.props.

Async actions

1) Ставимо axios та redux-thunk;

2) імпортимо applyMiddleware, compose з redux в store.js;

3) додаємо в createStore таку конструкцію compose(applyMiddleware(thunk), DevTools)(після rootReducer);

4) пишемо редюсер і екшн( в якому буде тип і параметр куди буде потрапляти інформація з фетча );

5) пишемо ф-ю яка робить запит і ретьорнить нам axios(проміс);

6) Пишемо асинхроний екшн: це ф-я яка викликає в середині себе ще одну ф-ю з параметром dispatch. В середині колбека запускаємо ф-ю з пункта №5, а в зені до неї активуємо за допомогою метода dispatch наш простий екшн з пункта №4;