AXIOS  
1. Подключение

npm i axios

2. Использоване

const res = await axios({url: '<https://jsonplaceholder.typicode.com/posts>'})

console.log(res, res.data);

Аксиос принимает один обьект со всеми настройками, и возвращает обьект со всеми данными и с полем data

Методы для упрощенной работы

axios.get(url)

axios.post(url[, data[, config]])

axios.isAxiosError(error) - показывает сгенерирована ли ошибка аксиомом или нет

3. Настройки запроса (все что передается в аргумент)

<https://axios-http.com/ru/docs/req_config>

baseURL - путь который можно установить как дефолтный, и дальше указывать в url только вторую часть

timeout: 1000 - прервать запрос если он превысит 1000 мсек

withCredentials: false - регулирует, отправлять ли куки авторизации (бэковские)

params - обьект в который передаем параметры и они потом переходят в квери-параметры

body - данные которые передаются для POST запроса

onDownloadProgress: function (progressEvent) {...} - обрабатывать степень загрузки данных

validateStatus: (status) => (status >= 200 && status < 300) - определять какой статус не error

proxy: - настройка прокси

4 Дефолтные настрйоки аксиоса (axios.defaults)

axios.defaults.baseURL = BASE\_URL

5. Дефолтные запросы axios instance (базовые настройки для работы с разными урлами или разными параметрами)

const firstRequest = axios.create({baseURL: '[https://jsonplaceholder.typicode.com](https://jsonplaceholder.typicode.com/)', withCredentials: true})

const secondRequest = axios.create({baseURL: '[https://222jsonplaceholder.typicode.com](https://222jsonplaceholder.typicode.com/)', withCredentials: false})

const firstData = firstRequest({url: 'posts'});

const secondData = secondRequest({url: 'images'});

6.Интерцепторы

срабатывает после получения ответа но перед дальнейшей его записью или обработкой

axios.interceptors.response.use((res) => {

console.log(res);

return res

}, (error) => {

console.log(error);

})

7. Отмена запроса

через abort()

-создаес контроллер для первого запроса

const firstController = new AbortController();

-в самом запросе в параметры прокидываем

signal: firstController .signal

- если хотим отменить - вызываем firstController.abort()