Реакт. Чек-лист для студентов

Работа отклоняется от проверки

- Пул-реквест не отправлен на проверку.
- Сборка или запуск проекта выполняются с ошибками.
- Не работает описанная в задании функциональность. API поиска, сохранения или удаления фильмов, а также авторизации, регистрации или редактирования профиля не работают или при обращении к ним возникают ошибки в консоли.
- Фронтенд-часть приложения не выложена на сервер.
- В файле README.md не приведена ссылка на макет, или макет по этой ссылке не открывается.
- Проект не соответствует сгенерированному под него макету.

Работа принимается, если соблюдены критерии работоспособности

- Функциональность проекта из брифа реализована полностью: `/ 35`
  - Общее:
    - Все ссылки и кнопки в проекте работают.
    - Правильно работают оба состояния шапки: если пользователь не залогинился, в шапке должны быть кнопки «Войти» и «Регистрация»; если пользователь залогинился, кнопки исчезают — и появляются кнопки «Фильмы», «Сохранённые фильмы» и «Аккаунт», в том числе и на главной странице.
    - При поиске текст запроса, найденные фильмы и состояние переключателя короткометражек сохраняются в хранилище. Если пользователь повторно переходит на страницу фильмов, то при монтировании компонента данные достаются из локального хранилища. Страница отображается в соответствии с загруженными из хранилища данными.
    - Все формы валидируются и на стороне клиента. Пользователь не может отправить запрос с невалидными данными.
  - Страницы «Регистрация» и «Авторизация»:
    - На странице «Регистрация» клик по кнопке «Зарегистрироваться» отправляет запрос на роут `/signup`, если данные введены корректно. Если запрос прошёл успешно, то автоматически производится вход и редирект на страницу `/movies`.
    - На странице «Авторизация» клик по кнопке «Войти» отправляет запрос на роут `/signin`, если данные введены корректно. Если запрос прошёл успешно, происходит редирект на страницу `/movies`.
    - Все формы валидируются и на стороне клиента.
  - Страница редактирование профиля:
    - На странице редактирования профиля клик по кнопке «Сохранить» отправляет запрос на роут `/users/me`, если данные введены корректно.
    - Пользователю отображается уведомление об успешном запросе к серверу при сохранении профиля.
    - Если на странице редактирования профиля введённая информация соответствует текущим данным пользователя, кнопка «Сохранить» заблокирована и нельзя отправить запрос сохранения.
  - Поиск фильмов:
    - Прелоадер крутится во время выполнения запроса фильмов.
    - Работа с фильтром настроена:
      - Поиск фильмов регистронезависимый.
      - Если запрос выполняется впервые, то работа с фильтром происходит после получения данных.
      - Если карточки уже были отображены на странице в блоке результатов, клик по чекбоксу «Короткометражки» приводит к повторной фильтрации результата.
    - После успешного сабмита формы поиска появляется блок с результатами.

        - Если ничего не найдено, выводится текст: «Ничего не найдено».
    - На странице всех фильмов в блоке результата отображается такое же количество карточек, как в макете. Нажатие на кнопку «Ещё» отображает следующий ряд с тем же числом карточек. При отображении всех карточек кнопка "Ещё" скрывается.
    - Карточки:
        - Карточка состоит из изображения, названия фильма и его длительности. Длительность фильма рассчитывается корректно и соответствует формату в макете. Клик по карточке ведёт на трейлер фильма.
        - Кнопка лайка имеет правильное состояние, в зависимости от того, добавлен ли фильм в сохранённые или нет.
        - При клике на иконку «Лайк» в блоке карточки выполняется запрос к `/movies` нашего API для установки или снятия лайка, в зависимости от текущего состояния.
    - На странице «Сохранённые фильмы»:
        - Отображается форма поиска. Она позволяет искать фильмы по уже полученным данным о сохранённых фильмах.
        - Блок карточки содержит кнопку удаления, а не лайка.
        - При нажатии на кнопку удаления выполняется запрос на удаление фильма. После успешного запроса карточка удаляется со страницы.
- Отсутствуют серьёзные баги, которые возникают при работе с сервисом, например: `/ 8`
    - Удалённые или добавленные карточки фильмов появляются только после перезагрузки страницы.
    - Если осталось отобразить меньше карточек фильмов, чем в полном ряду, то нажатие кнопки «Ещё» вызывает появление ошибок в консоли.
    - При удалении сохранённых карточек на соответствующей странице «Сохранённые фильмы» их по-прежнему можно найти через поиск. Поиск начинает корректно работать только после перезагрузки страницы.
    - Другие баги, которые возникают при работе с сервисом и требуют перезагрузки страницы или «ломают» пользовательский опыт.
- Регистрация и авторизация:
    - Роуты `/saved-movies`, `/movies`, `/profile` защищены HOC-компонентом `ProtectedRoute`. Роуты `/` , `/signin`, `/signup` не защищены. `/ 2`
    - При попытке перейти на любой защищённый роут происходит редирект на `/`. `/ 3`
    - Если пользователь был авторизован и закрыл вкладку, он может вернуться сразу на любую страницу приложения по URL-адресу, кроме страниц авторизации и регистрации. `/ 2`
    - После успешного вызова обработчика `onSignOut` происходит редирект на `/`. `/ 2`
    - Корректно используется хук `useHistory`. `/ 1`
    - При попытке перейти на несуществующую страницу происходит редирект на страницу «404». Кнопка «Назад» работает корректно. `/ 2`
    - Корректно используются компоненты `<Switch />`, `<Route />` и `<Redirect />`. `/ 1`
- Работа с JWT выполнена корректно:
    - JWT-токен хранится в `localStorage` или в `cookie`; `/ 2`
    - `Jwt` проверяется запросом к серверу, а не только в локальном хранилище. `/ 2`
    - При выходе из аккаунта `jwt` удаляется. `/ 3`
- Хуки:
    - Хуки не используются внутри условных блоков или циклов. `/ 1`
    - Хуки вызываются в основной функции компонента. `/ 1`
    - При использовании классовых компонентов эффекты описаны внутри методов жизненного цикла компонента. `/ 1`

    - Имена пользовательских хуков начинаются с `use`. `/ 2`
- Контекст:
    - В корневом компоненте `App` создана стейт-переменная, которая хранит данные пользователя. Она используется в качестве значения для провайдера контекста. `/ 1`
    - В компонент `App` внедрён контекст через `CurrentUserContext.Provider`. `/ 1`
    - Компоненты, в которых используются данные профиля, подписаны на контекст. `/ 1`
- Асинхронные запросы к API: ~~`/ 4`~~
    - Запросы можно осуществлять через Fetch API или XMLHttpRequest, сторонние библиотеки, такие как `axios` или `jQuery,` не применяются. `/ 1`
    - Запросы к API вынесены в отдельные файлы: `MainApi.js` и `MoviesApi.js`. `/ 1`
    - Первый обработчик `then` возвращает `res.json`. `res` проверяется на корректность. `/ 1`
    - Цепочка обработки промисов завершается блоком `catch`. `/ 1`
    - Не выполняются лишние запросы к бэкенду, например: запрос всех фильмов с сервиса beatfilm-movies производится только при первом поиске; все сохранённые фильмы не запрашиваются с сервера при каждом лайке или дизлайке; данные пользователя запрашиваются один раз при запуске приложения. `/ 2`
- Именование: `/ 2`
    - имена переменных, функций и параметров написаны в camelCase;
    - имена переменных — существительные;
    - имена переменных, функций и компонентов соответствуют содержимому;
    - имена коллекций NodeList — существительные во множественном числе;
    - имя функции начинается с глагола и отражает то, что она делает;
    - для именования запрещены: транслит и неуместные сокращения.

Хорошие практики

- Начальное состояние стейт-переменных содержит корректный тип данных. `/ 1`
- Для элементов списка используется уникальный ключ `key`, а не индекс массива. `/ 2`
- Запросы к API описаны внутри компонента `App` или в корневых компонентах страниц. `/ 1`
- Для внутренних ссылок в приложении используются компоненты из библиотеки `react-router`. `/ 2`
- Не происходит утечки памяти при навешивании обработчиков. Все обработчики, добавленные через `addEventListener`, удаляются при размонтировании компонента. `/ 2`
- Обработка ошибок API:
    - пользователь получает сообщение в случае любой ошибки; `/ 4`
    - поля формы заблокированы во время отправки запросов, и у пользователя нет возможности отправить новый запрос до завершения предыдущего. `/ 2`
- Фиксированные значения (константы) именуются заглавными буквами и вынесены в отдельный конфиг-файл. `/ 2`

Рекомендации

- Сторонние JavaScript-библиотеки не применяются. `/ 2`
- Используются семантически правильные блоки для компонентов. Не используются `<div>` или иные обертки для компонентов, которые состоят из одноуровневых блоков. `/ 1`
- Отсутствует «мусор» в коде `/ 2`:
    - нет беспорядка в коде;
    - нет лишнего кода: например, переменная объявлена, но не используется, или есть какая-то избыточная логика;
    - код форматирован одинаково, соблюдается иерархия отступов.