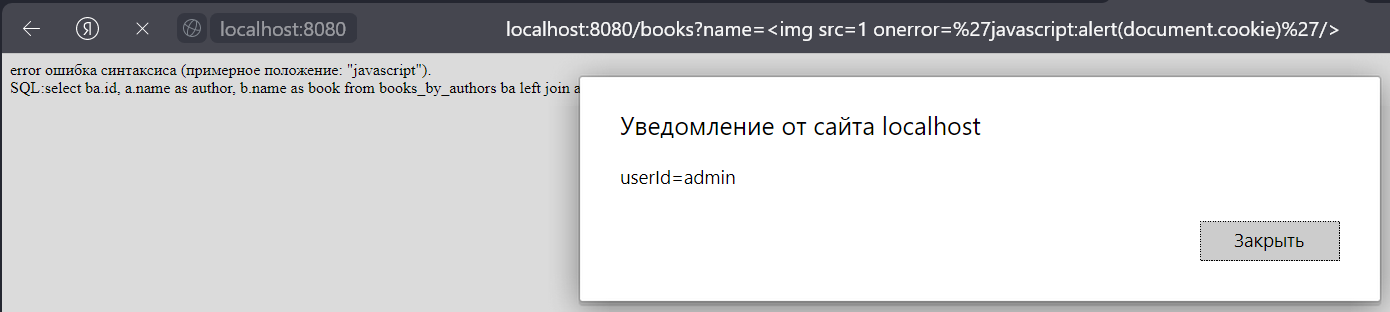
## Reflected XSS

Это простейшая разновидность межсайтового скриптинга. Они возникают, когда приложение получает данные в HTTP-запросе и включает эти данные в ответ небезопасным способом.

Используя строку поиска или напрямую через url, можно сформировать следующий запрос:

<img src=1 onerror='javascript:alert(document.cookie)'/>

Это позволит украсть cookie жертвы, перешедшей по ссылке.

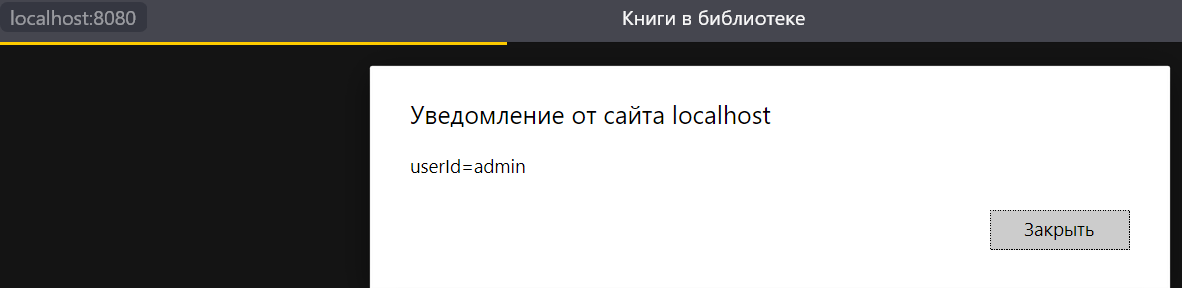


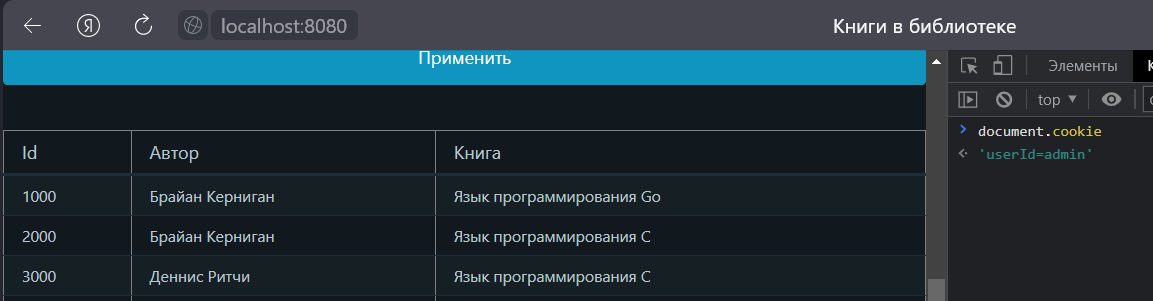
## Persisted XSS

Еще их называют хранимыми XSS. Возникают, когда приложение получает данные из ненадежного источника и небезопасным образом включает эти данные в свои последующие HTTP-ответы.

В строке названия добавляемой книги можно записать следующий скрипт, который будет выполняться у всех пользователей:

<script>alert(document.cookie);</script>

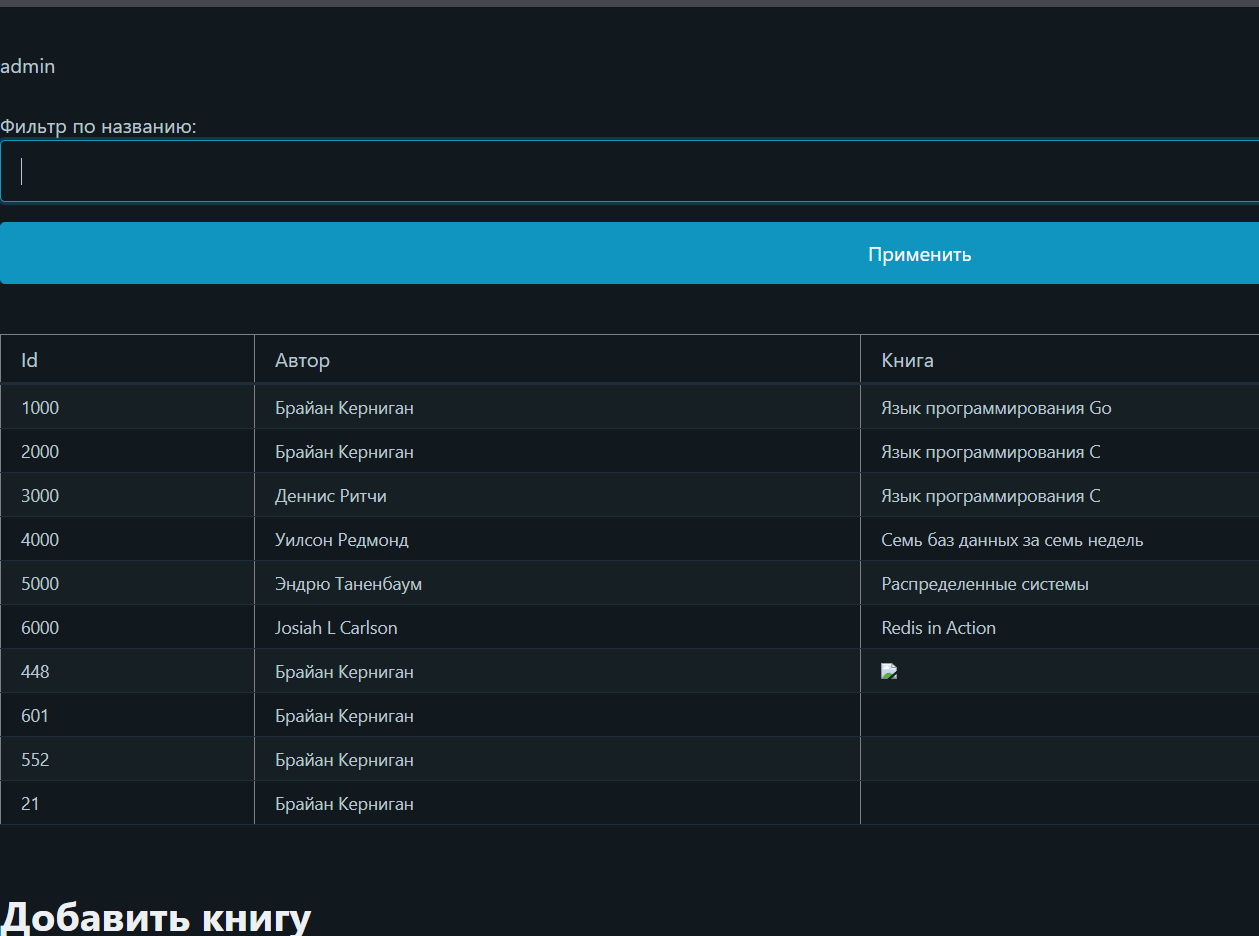




## Cookie Injection

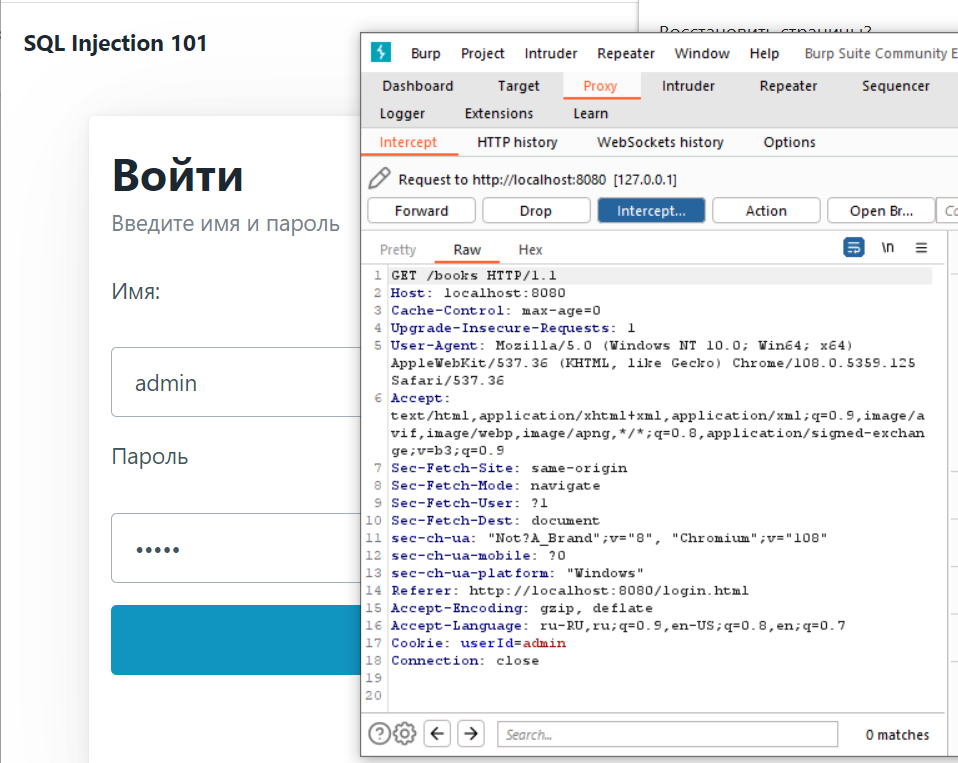
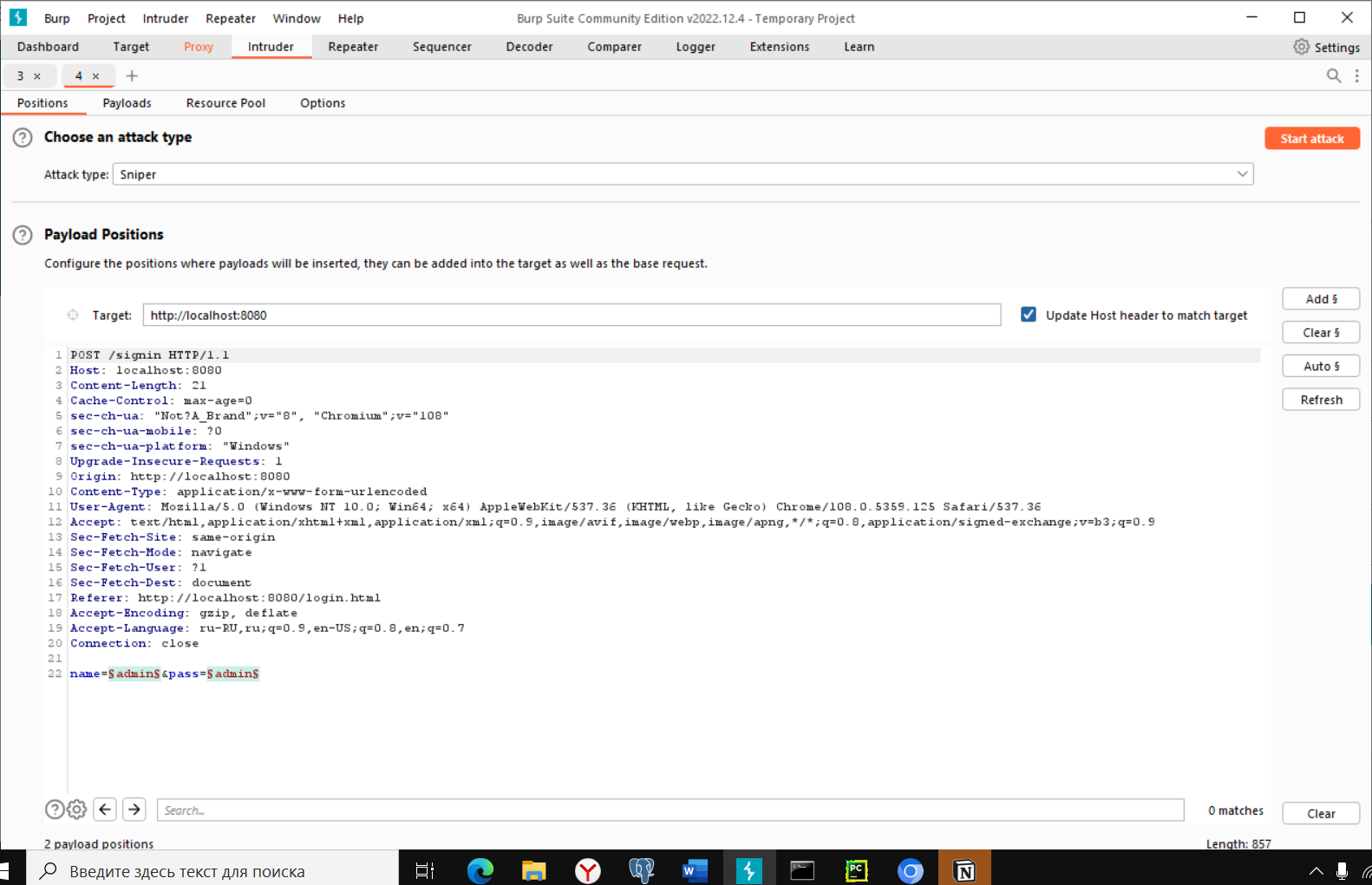
Кросс-доменная инъекция позволяет установить Cookie для запросов от *a.com* до *b.com.*

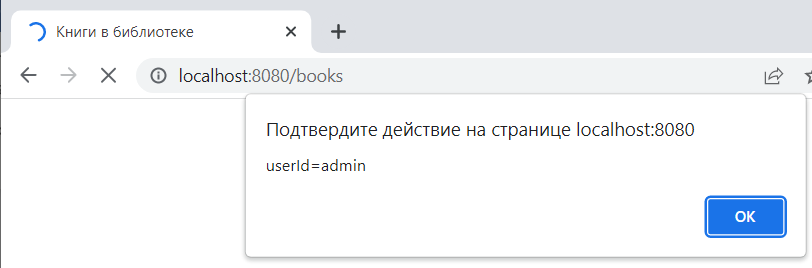
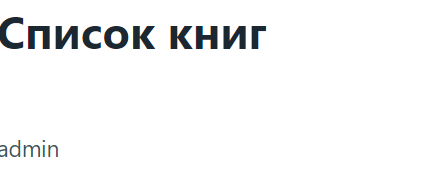
На сервер можно отправить запрос с хедером Cookie. Помимо подстановки значения cookie другого пользователя существует возможно подменить дополнительно установленные параметры.



## Session hijacking

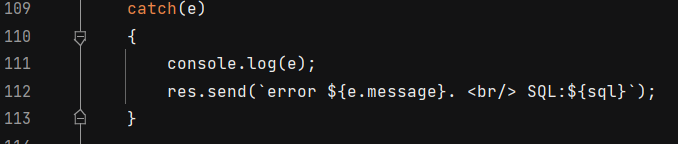
Пользователь в сеансе может быть захвачен злоумышленником и полностью потерять контроль над сеансом, при этом его личные данные могут быть легко украдены. После того, как пользователь начинает сеанс, например, заходит на банковский веб-сайт, злоумышленник может перехватить его. Так как на сервере не используется шифрование cookie могут быть перехвачены. Помимо этого, значение cookie может быть подобрано брут-форсом (в нашем случае он может осуществляться с помощью Intruder и заготовленного файла с паролями и логинами) или если алгоритм формирования значения недостаточно случайный.

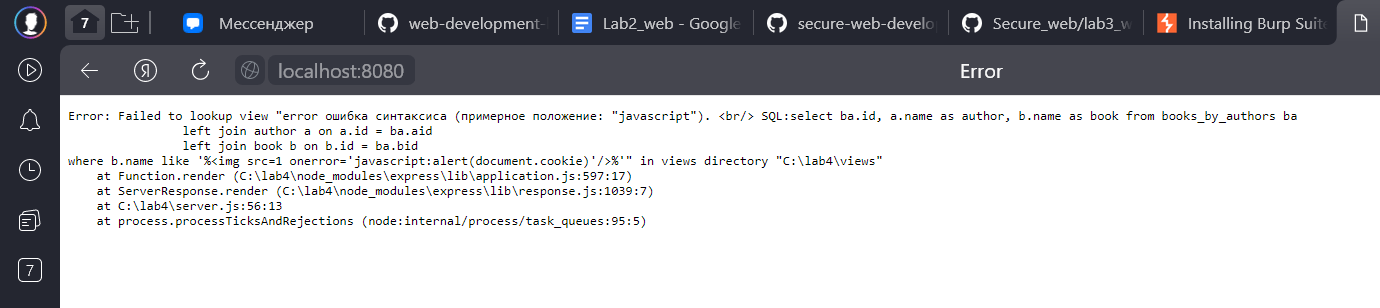




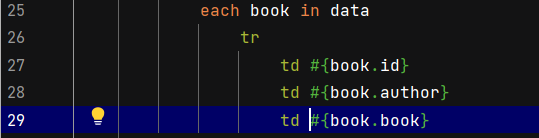
## Защита

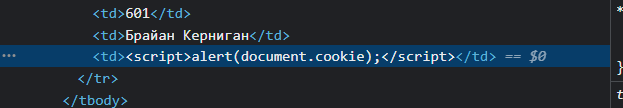
Для защиты от Reflected XSS следует доработать функцию обработки ошибок и заменить функцию res.send() на res.render() в файле server.js.



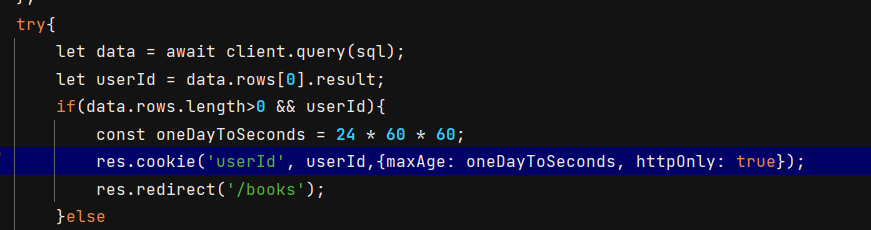


Для защиты от Persisted XSS следует доработать шаблон Pug и заменить знак «!» на «#».





Для защиты от cookie injection, а также для запрета доступа JavaScript к файлу cookie следует установить атрибут httpOnly в файле server.js . При этом, если бы сервер работал на HTTPS, то следовало бы установить атрибут Secure.



Проверяем защищенность, для этого проходим в панель разработчика и запускаем document.cookie