|  |
| --- |
| Stručni kurs Razvoj bezbednog softvera |
| Izveštaj |
| Pronađene ranjivosti u projektu “RealBookStore” |

|  |
| --- |
| Jovana Medenica  9-5-2025 |

# Istorija izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Verzija** | **Datum** | **Izmenio/la** | **Komentar** |
| **1.0** | 05.09.2025. | Jovana Medenica | SQL injecetion |
| **1.1** | 05.09.2025 | Jovana Medenica | Cross-site scripting |
| **1.2** | 05.09.2025 | Jovana Medenica | Cross-site request forgery |
| **1.3** | 05.09.2025 | Jovana Medenica | Authorization |
| **1.4** | 05.09.2025 | Jovana Medenica | Logging/Auditing |
|  |  |  |  |
|  |  |  |  |

Sadržaj

[Istorija izmena 1](#_Toc160050508)

[Uvod 3](#_Toc160050509)

[O veb aplikaciji 3](#_Toc160050510)

[Kratak pregled rezultata testiranja 3](#_Toc160050511)

[SQL injection 4](#_Toc160050512)

[Napad: Ubacivanje novog usera u tabelu “persons” (SQL injection) 4](#_Toc160050513)

[Metod napada: 4](#_Toc160050514)

[Predlog odbrane: 4](#_Toc160050515)

[Cross-site scripting 5](#_Toc160050516)

[Napad: Ubacivanje novog usera u tabelu “persons” 5](#_Toc160050517)

[Metod napada: 5](#_Toc160050518)

[Predlog odbrane: 5](#_Toc160050519)

[Zaključak 6](#_Toc160050520)

# Uvod

Ovaj izveštaj se bavi ranjivostima pronađenim u dole opisanoj veb aplikaciji.

## O veb aplikaciji

RealBookStore je veb aplikacija koja pruža mogućnosti pretrage, ocenjivanja i komentarisanja knjiga.

Aplikacija RealBookStore omogućava sledeće:

* Pregled i pretragu knjiga.
* Dodavanje nove knjige.
* Detaljan pregleda knjige kao i komentarisanje i ocenjivanje knjige.
* Pregled korisnika aplikacije.
* Detaljan pregled podataka korisnika.

## Kratak pregled rezultata testiranja

*Ovde idu kratko opisani rezultati testiranja: pronađene ranjivosti i nivo opasnosti.*

|  |  |
| --- | --- |
| ***Nivo opasnosti*** | ***Broj ranjivosti*** |
| ***Low*** | *3* |
| ***Medium*** | *2* |
| ***High*** | *1* |

# **SQL injection**

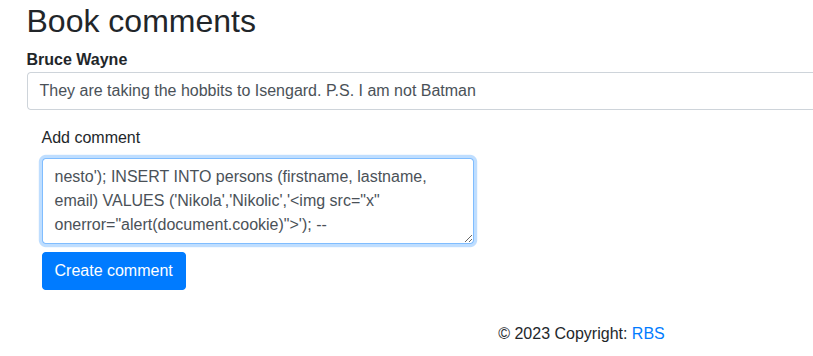
## Napad: Ubacivanje novog usera u tabelu “persons” (SQL injection)

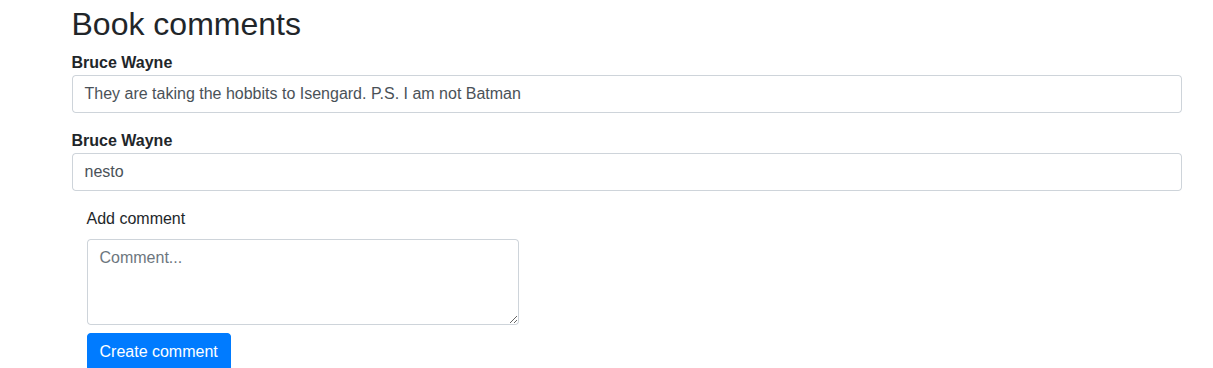
### Metod napada:

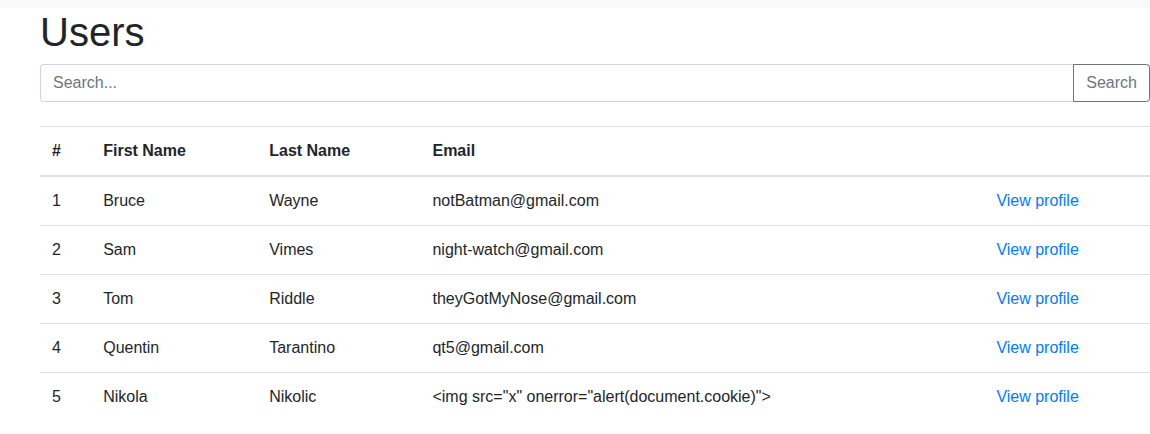
Na stranici /books/{id} aplikacije uneti sledeći kod u polje za ostavljanje komentara:

comment’); insert into persons(firstName,lastName,email)

values(‘Nikola’,’Nikolic’,’<img src=”x” onerror=”alert(document.cookie)”>’);--

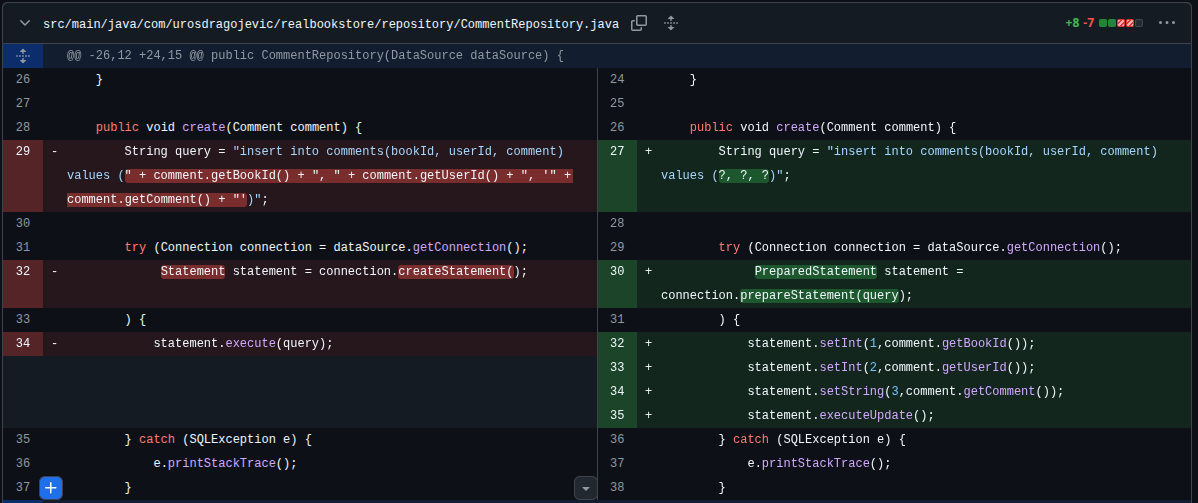
Možemo primetiti da je komentar unet I ne vidi se zli SQL upit.

Ali ono što se stvarno desilo je problem. Naime, sada je dodat novi user pomoću ovog upita. Što možemo videti ako odemo na tu stranicu.



## Predlog odbrane:

Ovo možemo da izbegnemo ukoliko umesto Statement klase čuvamo komentare koristeći klasu PreparedStatement. S leve strane se nalazi stara implementacija, a s desne je implementirana odbrana od SQL injection iz komentara.



# Cross-site scripting

## Napad: Ubacivanje novog usera u tabelu “persons”

### Metod napada:

Na stranici /persons aplikacije unet je sledeći kod u input polje “Email” kroz SQL injection napad:

‘<img src=”x” onerror=”alert(document.cookie)”>’

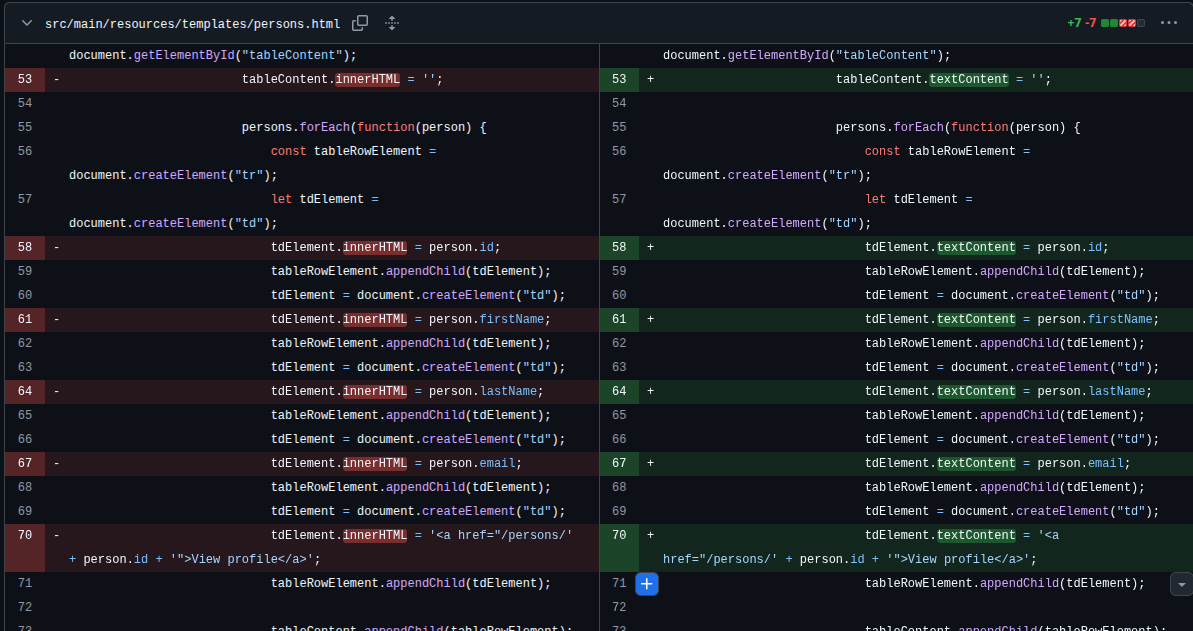
Nakon što se u pretrazi ukuca ime Nikola,

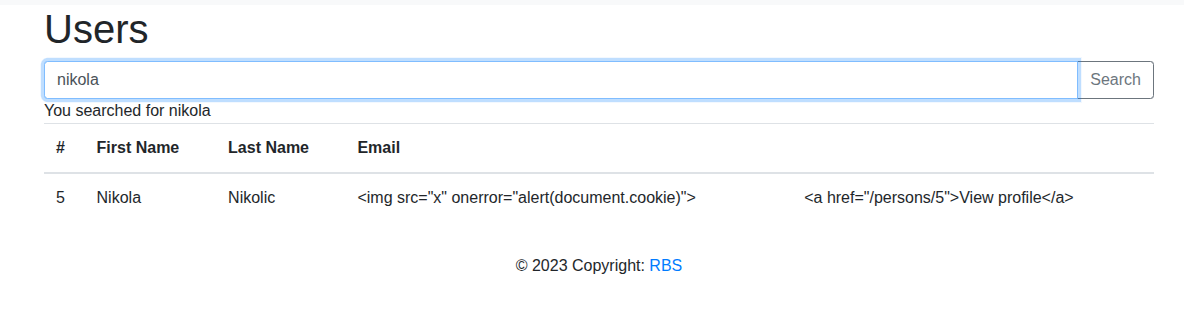
## 

## Pošto je postavljeno da su cookie HttpOnly, zvanično ne može da se pristupi tim informacijama preko document.cookie, ali ukoliko izmenimo da nije HttpOnly dobijamo cookie u celosti preko alert ekrana, kao na slici iznad.

## Predlog odbrane:

Kod pretrage umesto korišćenja innerHtml koristi se textContent za popunjavanje vrednosti u poljima.

Kao rezultat pretrage, nakon ove promene, dobijamo sledeći rezultat:

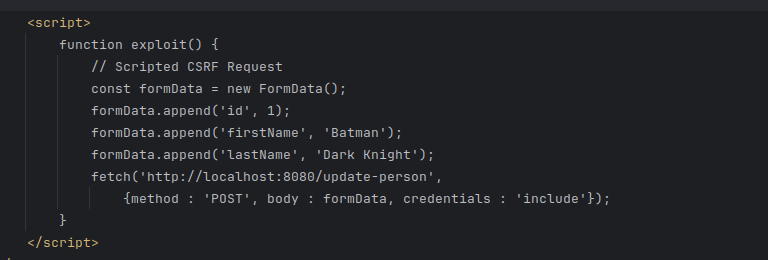


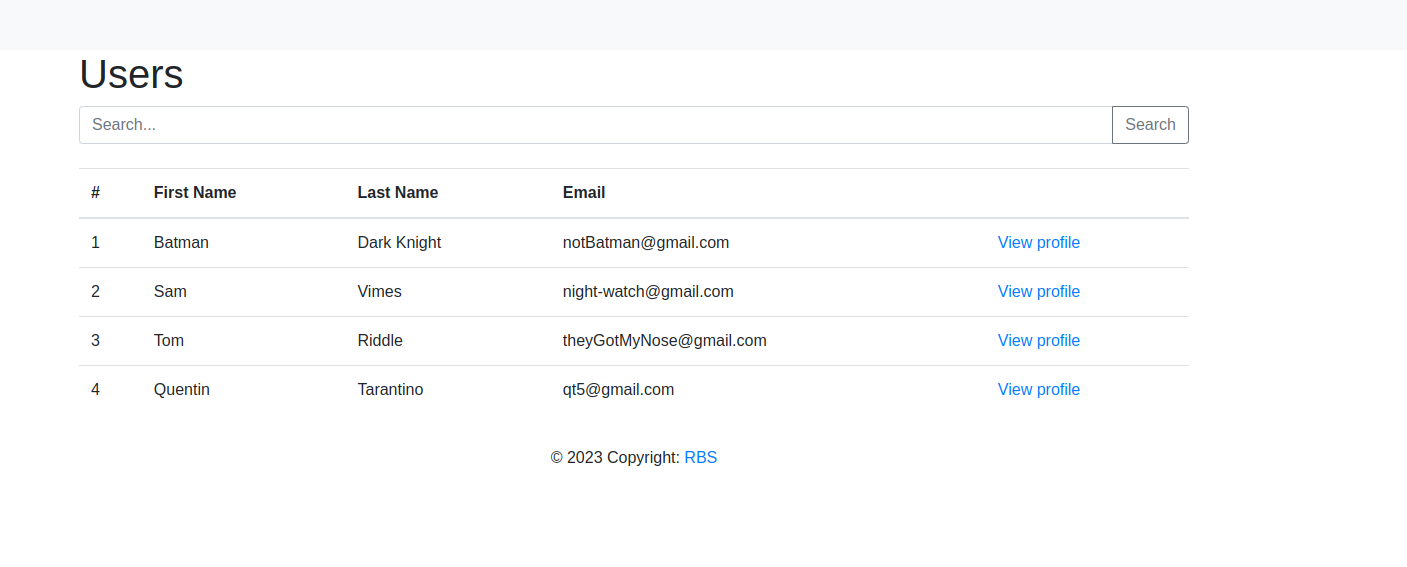
# Cross-site request forgery

## Napad: Promena podataka korisnika

### Metod napada:

Postoji maliciozna aplikacija na stranici /3000 na kojoj se nalazi pehar I klikom na njega “dobija se nagrada”, a zapravo se šalju podaci na /update-person I izmenjeni su ime I prezime korisnika sa id 1.

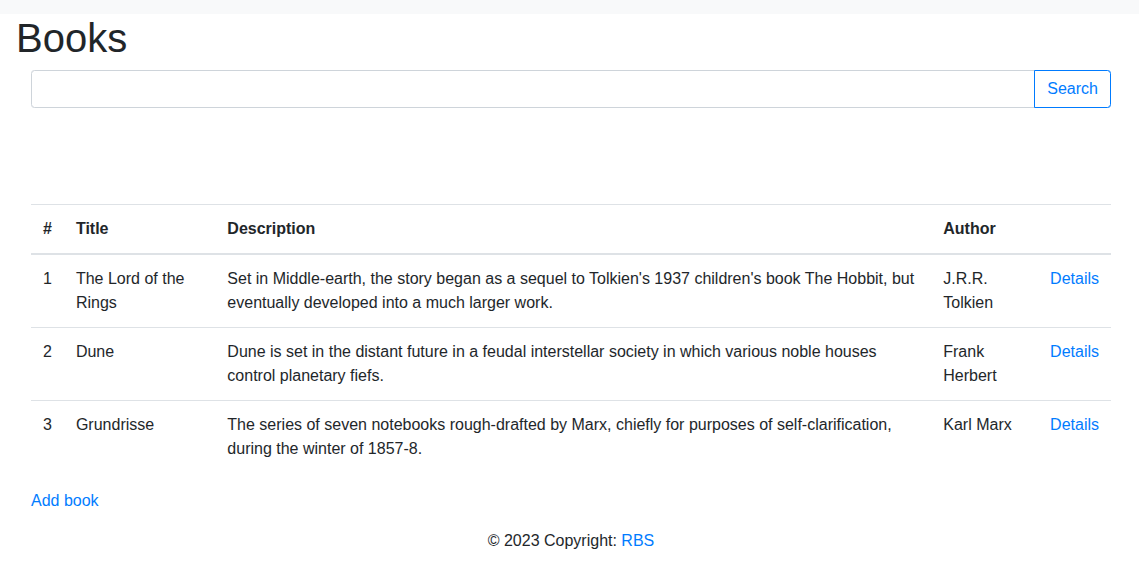
Dobije se sledeće nakon napada:

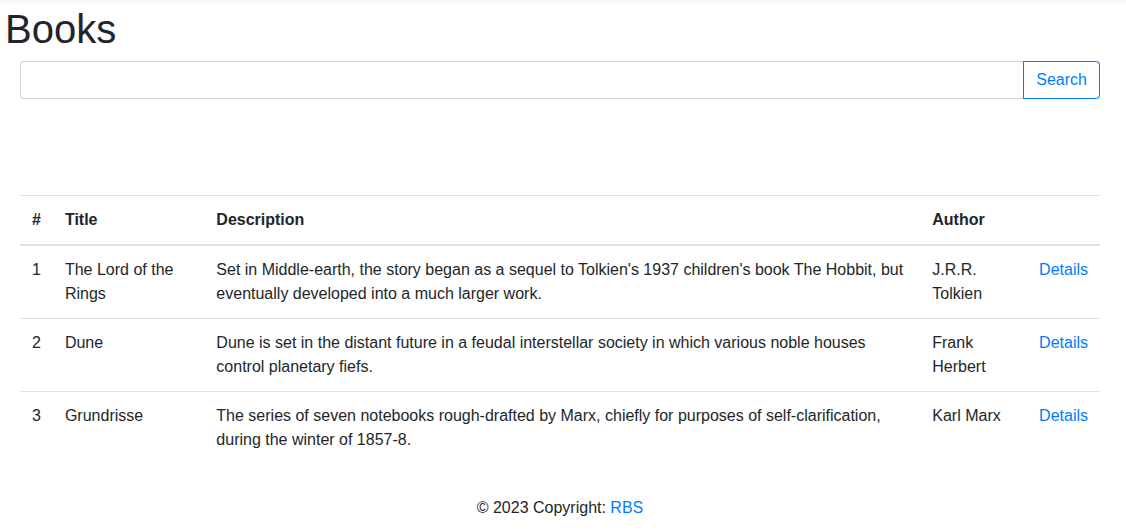
Predlog odbrane:

Implementacija CSRF zaštite pomoću tokena. Pri kreiranju forme dodaje se skriveno dugme csrfToken, a kada se pošsalje zahtev za izmenom proverava se token I ukoliko je validan onda se menjaju podaci, u suprotnom sve ostaje nepromenjeno.

# Authorization

## Neophodno je da odredimo ko ima koja prava prilikom korišćenja naše aplikacije. Zato se u bazi podataka uvodi autorizacioni model. Zatim učitavamo permisije I proveravamo ih prilikom pristupa podacima. Na sledećim slikama možemo videti kako aplikacija izgleda za dva korisnika koji imaju drugačija prava.





Razlika je u tome što prvi korisnik može da doda knjigu, a drugi ne može.

# DevOps – logging and auditing

## Koristili smo logging I auditing da bi mogli da obradimo greške u našem programu. Pomoću njih će programerima biti lakše da vide kada se neki problem desi, kao I da vide u čemu je tačno problem.

# Zaključak

## Usled nepažnje prilikom implementacije došlo je do stvaranja propusta koji mogu biti jako opasni po našu aplikaciju. Sreća to su sve to neke stinice na koje sledeći put treba obratiti pažnju. Pogotovo na sličnosti određenih metoda, koji na prvi pogled rade istu stvar, a zapravo nije baš tako.