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

|  |
| --- |
| Mateja Miletić  10.09.2025. |

# Istorija izmena

|  |  |  |  |
| --- | --- | --- | --- |
| **Verzija** | **Datum** | **Izmenio/la** | **Komentar** |
| **1.0** | 10.09.2025. | Mateja Miletić | Kreiran izveštaj |
| **1.1** | 10.09.2025. | Mateja Miletić | SQL injecetion |
| **1.2** | 10.09.2025. | Mateja Miletić | Cross-site scripting |
| **1.3** | 10.09.2025. | Mateja Miletić | Cross-site request forgery |
| **1.4** | 10.09.2025. | Mateja Miletić | Authorization |
| **1.5** | 10.09.2025. | Mateja Miletić | DevOps |
|  |  |  |  |

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](#_Toc160050516) 6

[Napad: Ubacivanje novog usera u tabelu “persons”](#_Toc160050517) 6

[Metod napada:](#_Toc160050518) 6

[Predlog odbrane:](#_Toc160050519) 8

Cross-site request forgery9

[Metod napada:](#_Toc160050518) 9

[Predlog odbrane:](#_Toc160050519) 11

Authorization12

DevOps13

[Zaključak](#_Toc160050520) 13

# 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.*

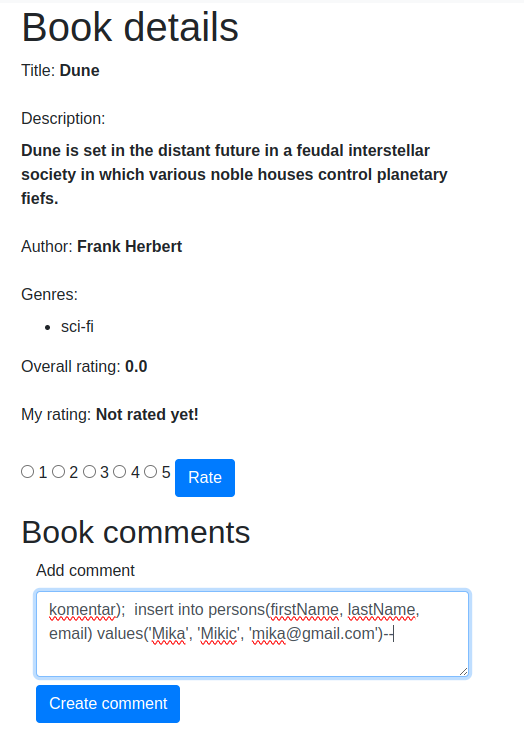
|  |  |
| --- | --- |
| ***Nivo opasnosti*** | ***Broj ranjivosti*** |
| ***Low*** | *3* |
| ***Medium*** | *2* |
| ***High*** | *1* |

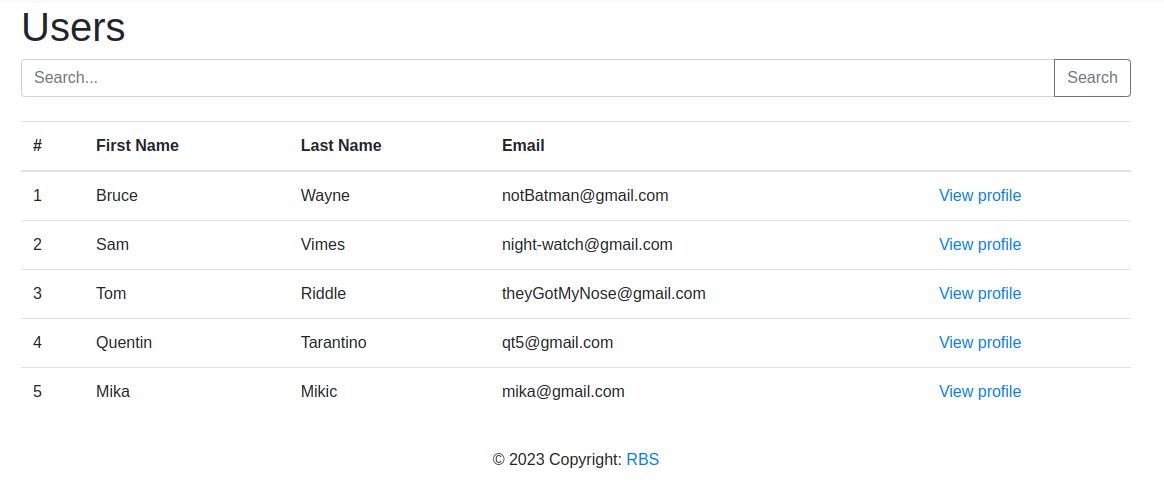
# SQL injection

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

### Metod napada:

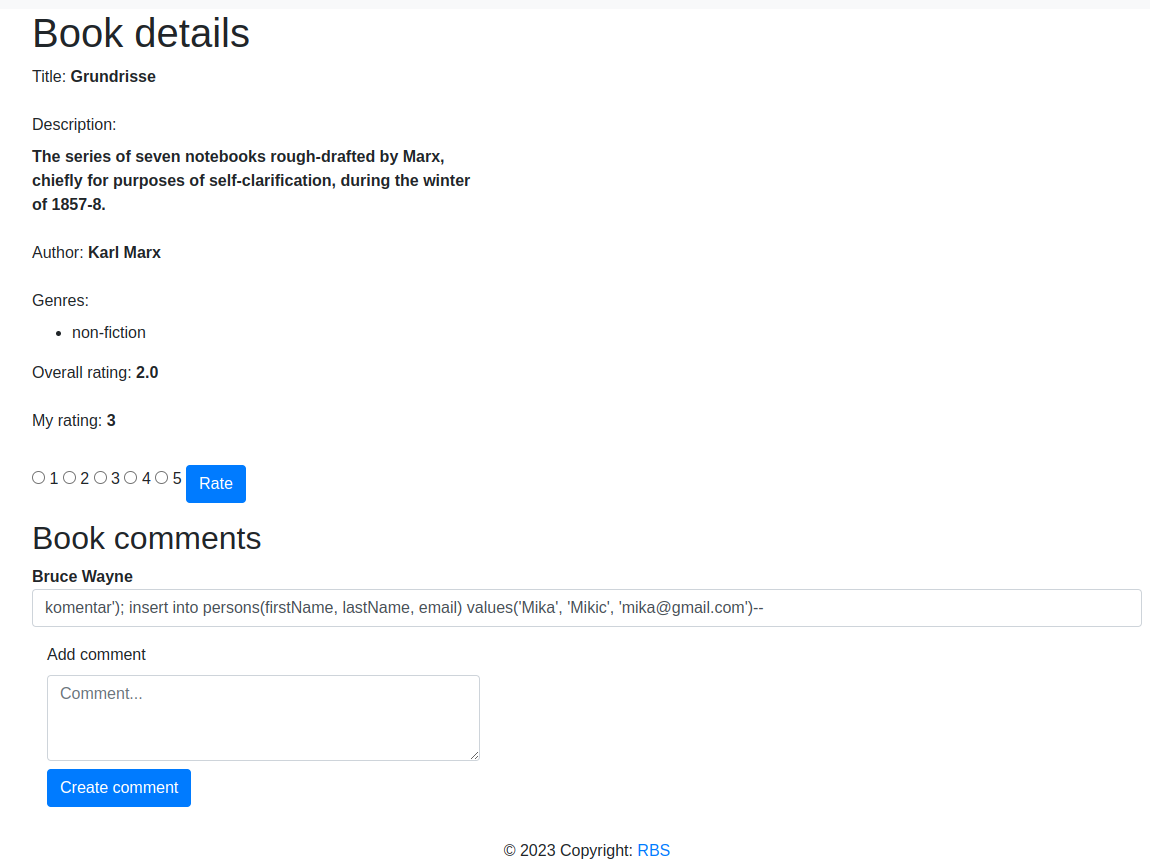
Izvršavamo napad unošenjem upita u polje "Book comments". Ovim smo, pored komentara "Komentar", dodali i korisnika u tabelu "Persons" kroz polje "Book comments".





## Predlog odbrane:

Od ovakvog napada se branimo tako što menjamo objekat klase Statement objektom klase PreparedStatement. Nakon odbrane komentar ce idalje biti dodat, ali novi korisnik nece.



# 

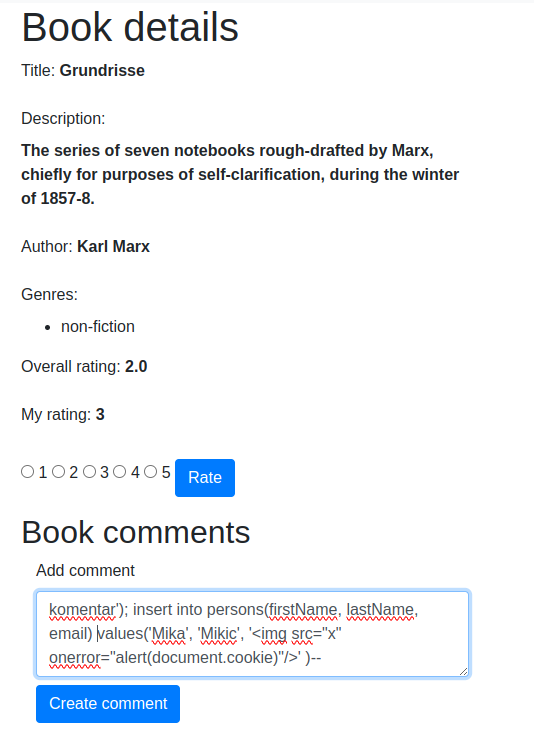
# Cross-site scripting

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

### Metod napada:

Izvršavamo Cross-site scripting napad unošenjem upita u polje "Book comments". Ovim smo, pored komentara dodali i korisnika u tabelu "Persons" kroz polje "Book comments". Za razliku od malopre

sada umesto email adrese unosimo JavaScript kod.



## 

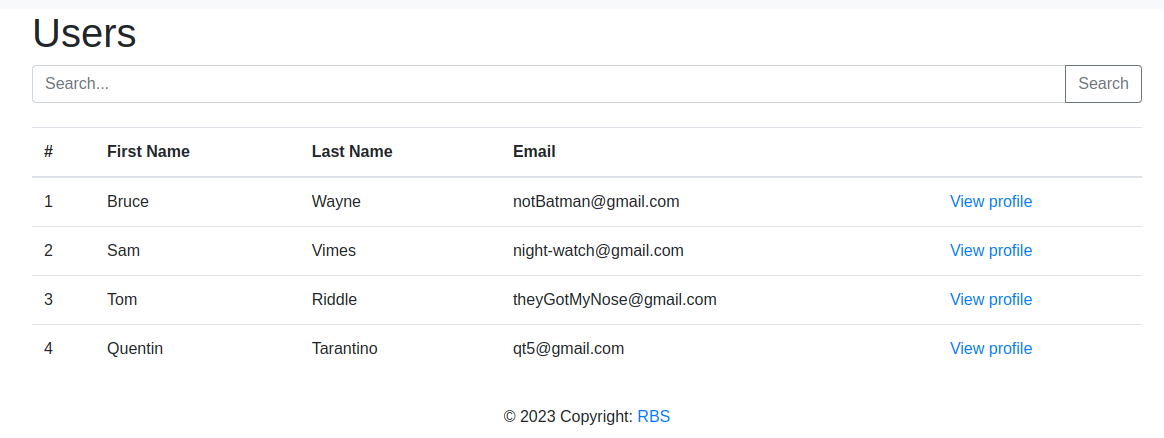
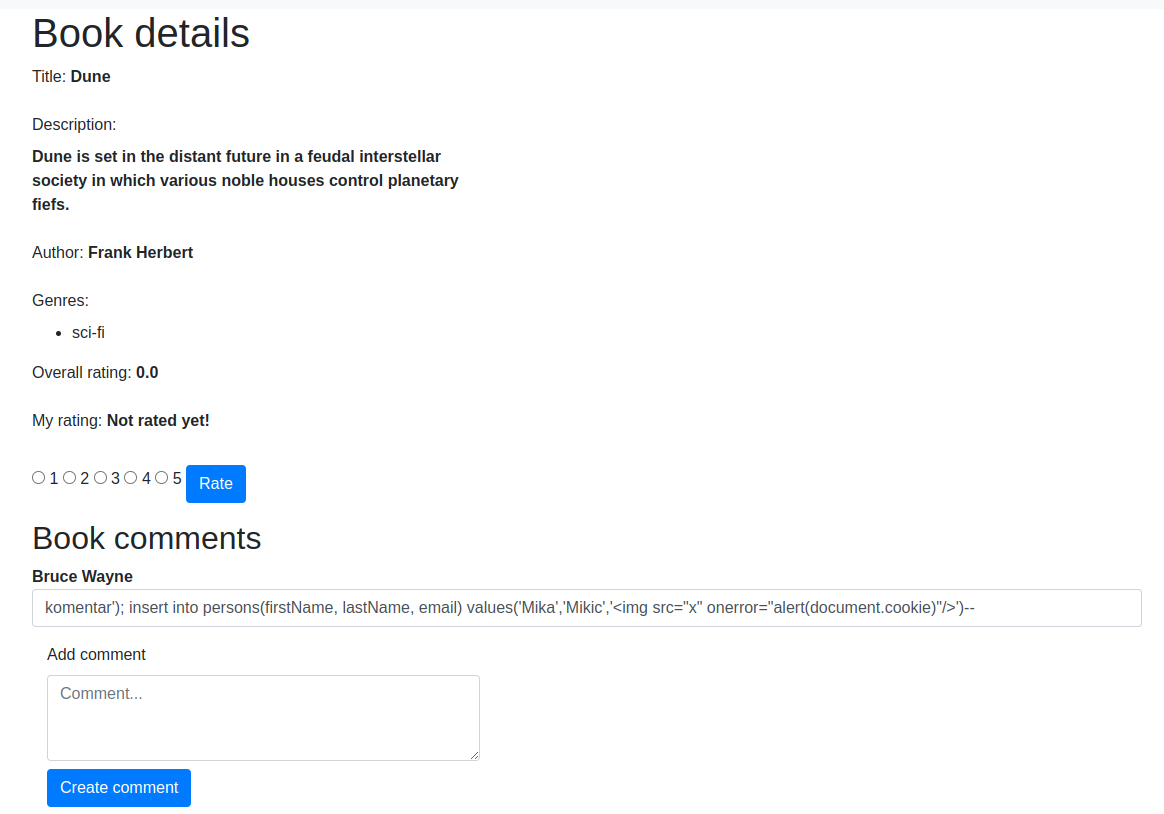
## Ukoliko pokušamo da pretražimo datog korsnika, biće nam prikazan alert.

## 

## Predlog odbrane:

Od napada se branimo tako što zamenimo objekat klase Statement objektom klase PreparedStatement. Pored toga potrebno je u persons.html zameniti innerHTML atribut textContent atributom, a u book.html fajlu th:utext sa th:text. Nakon ovih promena zaštićeni

smo od Cross-site scripting napada, samo će komentar biti dodat, dok novi korisnik neće.



# Cross-site request forgery

### Metod napada:

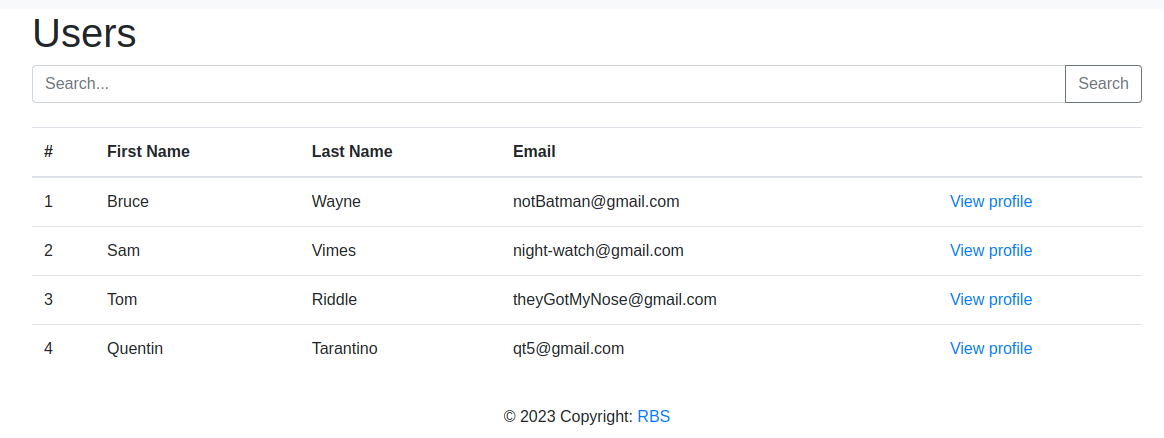
U sklopu csrf-exploit direktorijuma nalazi se index.html fajl i u njemu je implementirana funkcija exploit().

# 

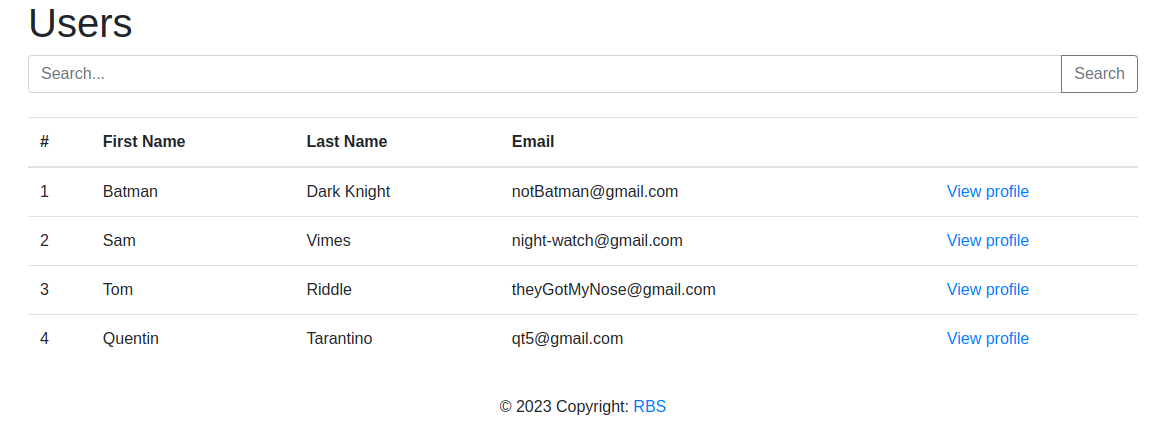
# Nakon implementacije ove funkcije u terminalu se pozicioniramo u csrf-exploit direktorijum i komandom npm start pokrećemo napadačev sajt na adresi http://localhost:3000.

# 

# Pritiskom na pehar u pozadini se šalje zahtev koji menja imena korisnika čiji je id = 1.Spisak korisnika pre nego što izvršimo napad:

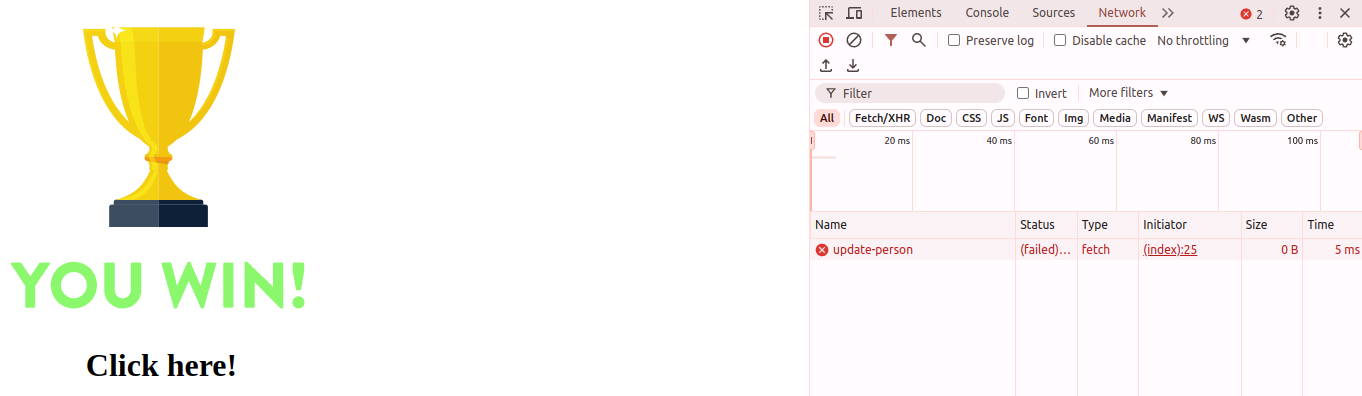


Spisak korisnika nakon što izvršimo napad:

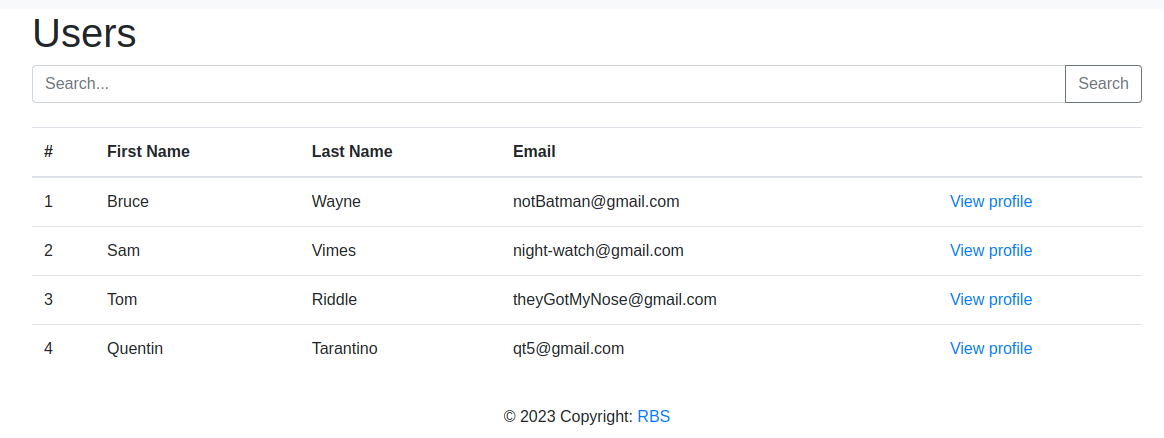


## Predlog odbrane:

Odbranu vršimo tako što u klasi PersonController i metodi person čitamo token iz sesije i upisujemo ga u model, a u metodi updatePerson dohvatamo vrednost CSRF tokena koji je poslat sa formom i upoređujemo ga sa onim koji se nalazi u sesiji. U formi za promenu detalja korisnika, dodajemo input element koji će sadržati vrednost CSRF tokena. Sada ukoliko pokušamo da izvršimo napad, klikom na pehar biće nam prikazano sledeće:

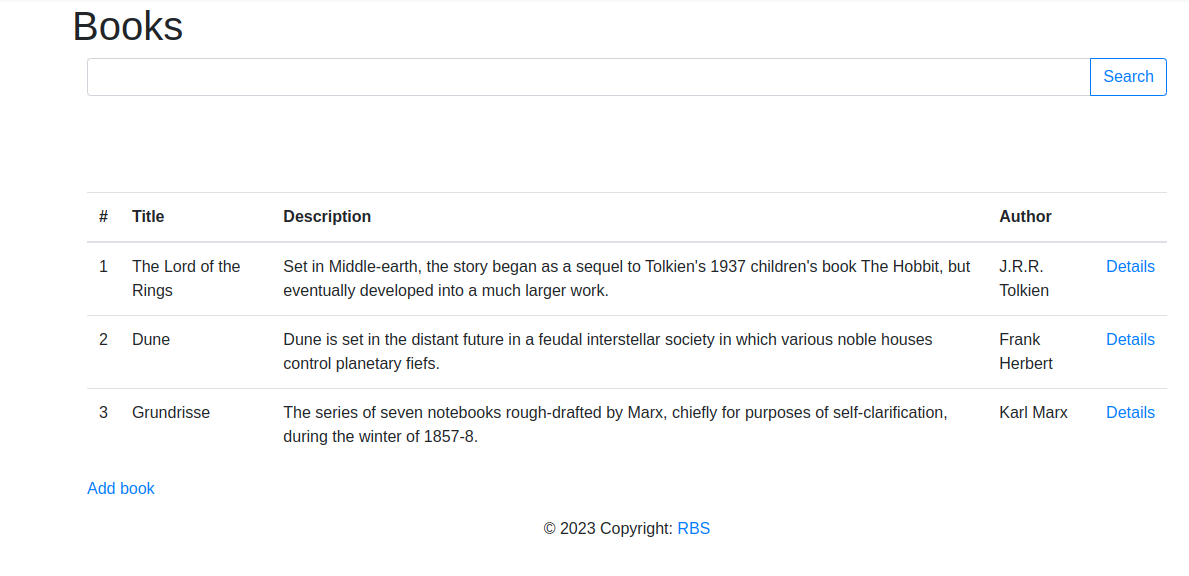


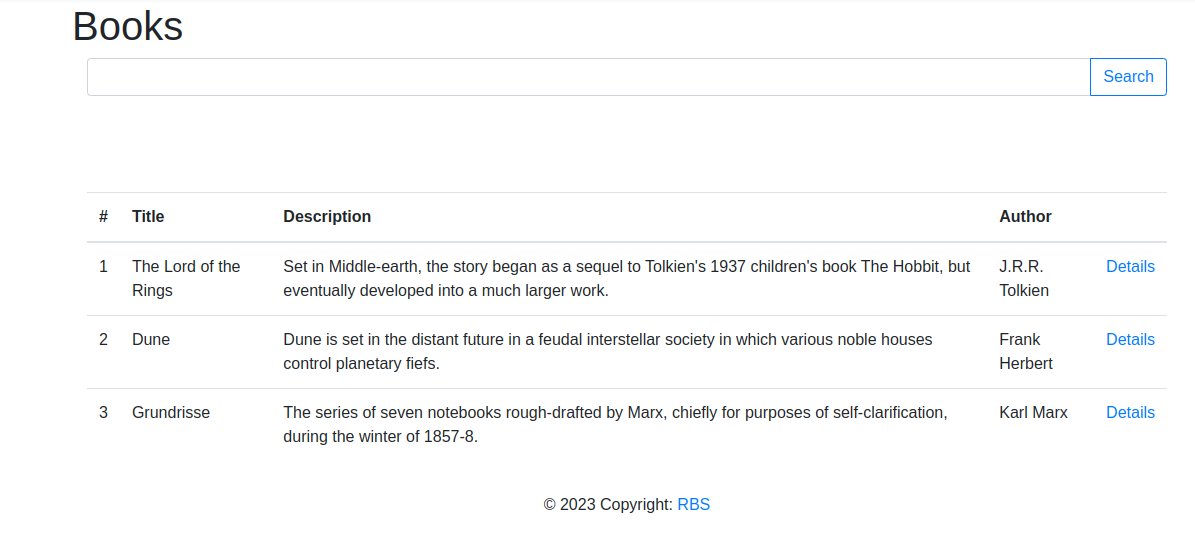
# Takođe i spisak korisnika će ostati nepromenjen.



Authorization

Potrebno je jasno definisati koji korisnici imaju određena prava u korišćenju aplikacije. Zbog toga u bazu podataka uvodimo model za autorizaciju. Nakon toga učitavamo dozvole i proveravamo ih prilikom pristupa resursima. Na narednim slikama prikazana je aplikacija iz ugla dva korisnika sa različitim privilegijama. Razlika između njih je u tome što prvi korisnik ima mogućnost dodavanja nove knjige, dok drugi tu opciju nema.





DevOps

U sistem smo uveli logging i auditing kako bismo olakšali detekciju i analizu grešaka. Na ovaj način programeri mogu jednostavnije da uoče kada je do problema došlo i da preciznije identifikuju njegov uzrok.

# Zaključak

Do napada je došlo zbog propusta u kodu koji su se mogli relativno jednostavno sprečiti. Preporuka je da se izvrši dodatna provera ostatka projekta kako bi se utvrdilo da li postoje slične ranjivosti.