Stručni kurs Razvoj bezbednog softvera

Izveštaj

Pronađene ranjivosti u projektu “RealBookStore”

Luka Arambašić 169/2020 Matematički fakultet

Sadržaj

[Uvod 1](#_Toc1269908671)

[O veb aplikaciji 1](#_Toc700700755)

[Kratak pregled rezltata testiranja 1](#_Toc1562252794)

[SQL Injection 1](#_Toc411571854)

[Napad: Ubacivanje novog korisnika u tabelu “persons” (SQL Injection) preko polja za komentar 1](#_Toc78774782)

[Odbrana: korišćenje parametara umesto direktne konkatenacije 1](#_Toc1176907818)

[SQL Injection + XSS (Cross-site scripting) 1](#_Toc1311334364)

[Napad: Ubacivanje maliciozne skripte (XSS) u jedan od atributa novoubačenog korisnika koji se ubacuje preko SQL Injection-a kroz polje za pretragu osoba 1](#_Toc1584128507)

[Odbrana: Za SQL injection identično, otklanjamo direktnu konkatenaciju, a za XSS menjamo .innerHtml sa .textContent 1](#_Toc1390949183)

[Cross-site request forgery 1](#_Toc395943634)

[Napad: Navodimo žrtvu da klikne link koji šalje zahtev koji menja podatke 1](#_Toc1116449607)

[Odbrana: Implementiramo logiku tokena 1](#_Toc2141311893)

# Uvod

Ovaj izveštaj se bavi ranjivostima u dole opisanoj veb aplikaciji.

## O veb aplikaciji

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

Aplikacija RealBookStore omogućava sledeće:

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

## Kratak pregled rezltata testiranja

Na projektu je korišćen alat SonarQube za statičku analizu koda i pronalaženje ranjivosti.

Pronađeno je 20 ranjivosti.

19 ranjivosti su SQL injection od kojih je samo jedna lažna pozitivna.

1 ranjivost je CSRF

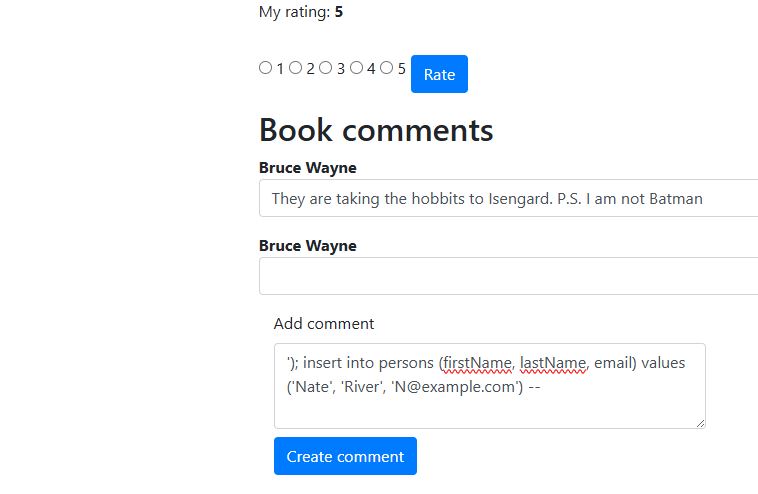
Alat je svim ranjivostima dodelio “High” nivo opasnosti.

# SQL Injection

## Napad: Ubacivanje novog korisnika u tabelu “persons” (SQL Injection) preko polja za komentar

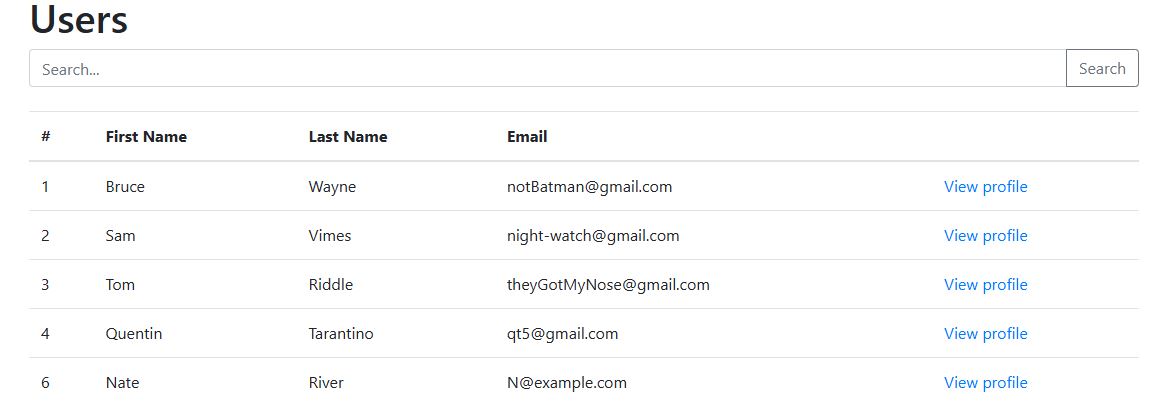
Na stranici za unošenje komentara uneti sledeći kod u polje za komentar:

'); INSERT INTO persons (firstName, lastName, email) VALUES ('Nate', 'River', N@example.com'); --

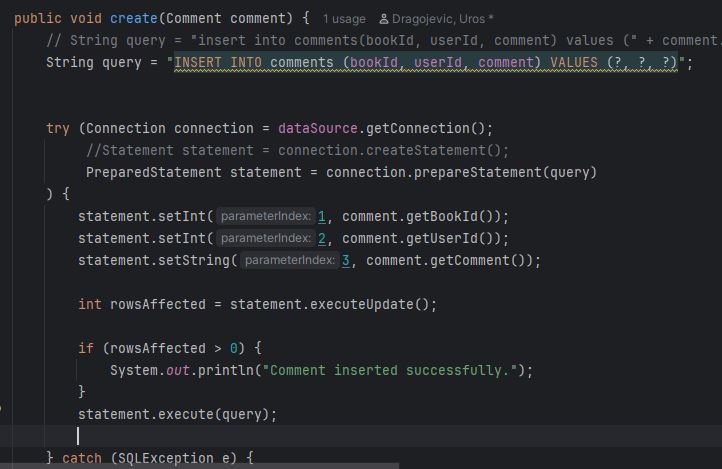


Kod unosi proizvoljnog korisnika u tabelu “persons”.

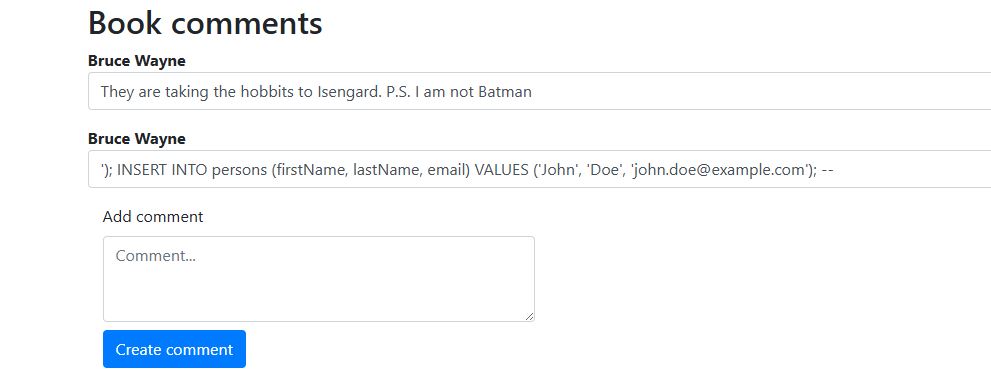
Nakon toga se na stranici “persons” može uočiti dodat novi korisnik:



## Odbrana: korišćenje parametara umesto direktne konkatenacije



Nakon izmenjenog koda i ponovnog pokušaja napada možemo uočiti da ne dolazi do neželjenog ponašanja naše aplikacije, već komentar ostaje u formi u kojoj je unet

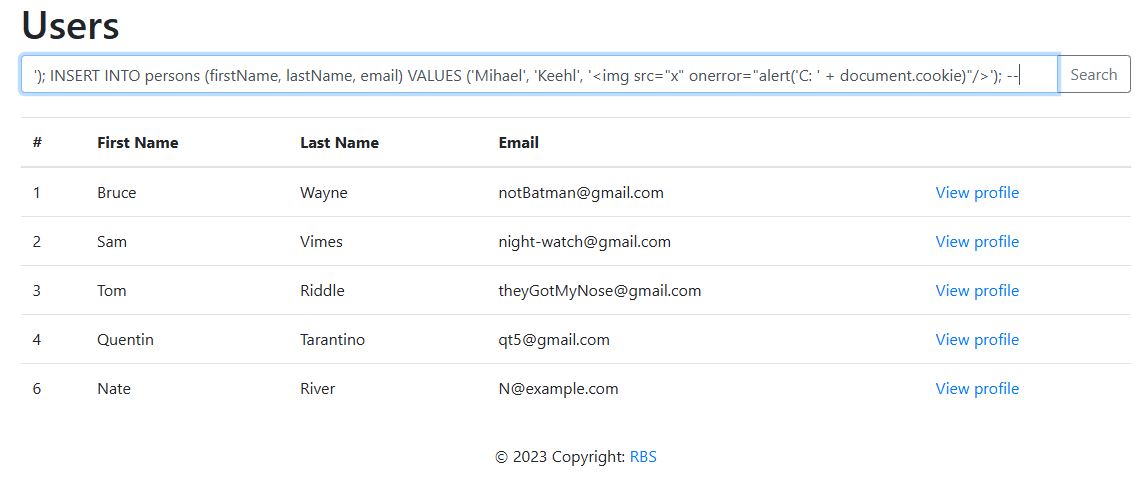


# SQL Injection + XSS (Cross-site scripting)

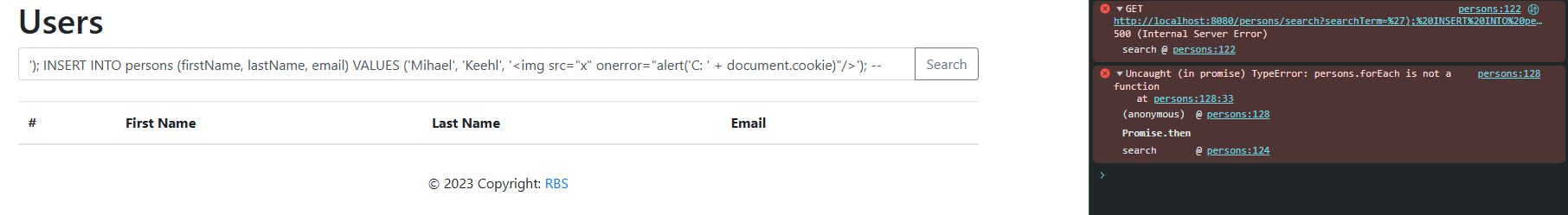
## Napad: Ubacivanje maliciozne skripte (XSS) u jedan od atributa novoubačenog korisnika koji se ubacuje preko SQL Injection-a kroz polje za pretragu osoba

U polje za pretragu se unosi sličan kod kao za prošli napad, sa dodatkom da ćemo umesto email adrese kao atribut novog korisnika dodati kod za skriptu koja će u konzoli razvojnog alata ispisati cookie ID:

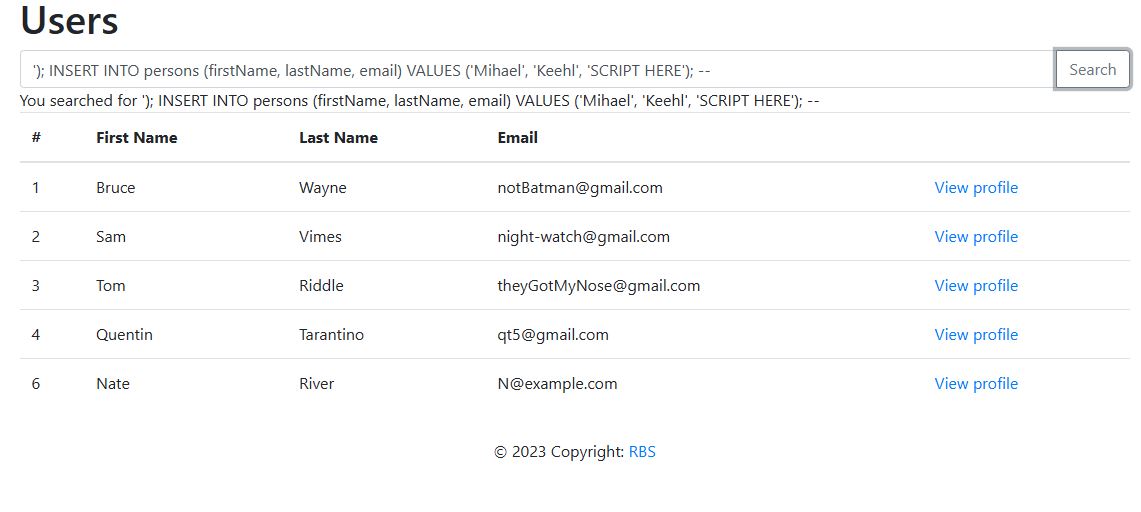
'); INSERT INTO persons (firstName, lastName, email) VALUES ('Mihael', 'Keehl', '<img src="x" onerror="console.log('Stolen Cookie:', document.cookie)"/>'); --

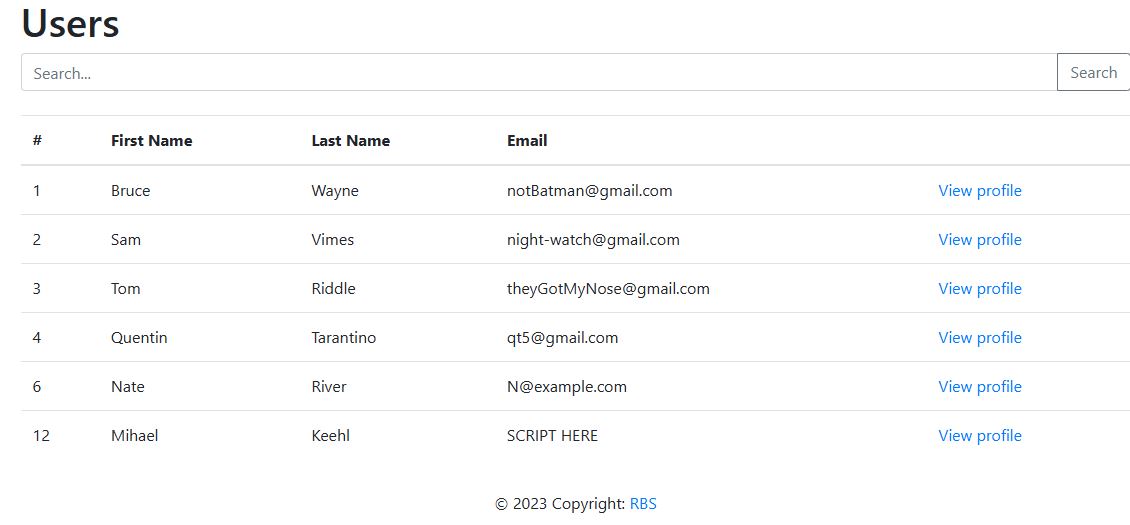


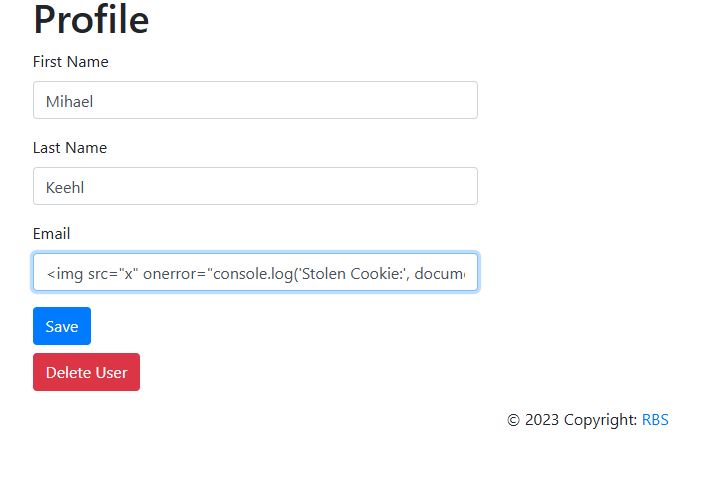
Pritiskom na dugme search bi trebalo da se korisnik ubaci u bazu i ponovo da se aktivira pretraga kako bi se aktivirala skripta. Ali nailazi se na problem. Ne dozvoljava se ubacivanje takvog koda u korisničko polje. To može biti iz raznih razloga jer posoje odbrane na više nivoa od XSS napada koje su veoma česta pojava. Greška koja izlazi u konzoli:



Ali ne treba odustajati! Primetimo da se ova interakcija može izvršiti u dva koraka s obzirom da imamo mogućnost da kliknemo na dugme “View profile” i da naknadno izmenimo vrednosti atributa za željenog korisnika. Korisnika ćemo uneti preko SQL Injection-a a naknadno ćemo u polje za email ubaciti našu skriptu:

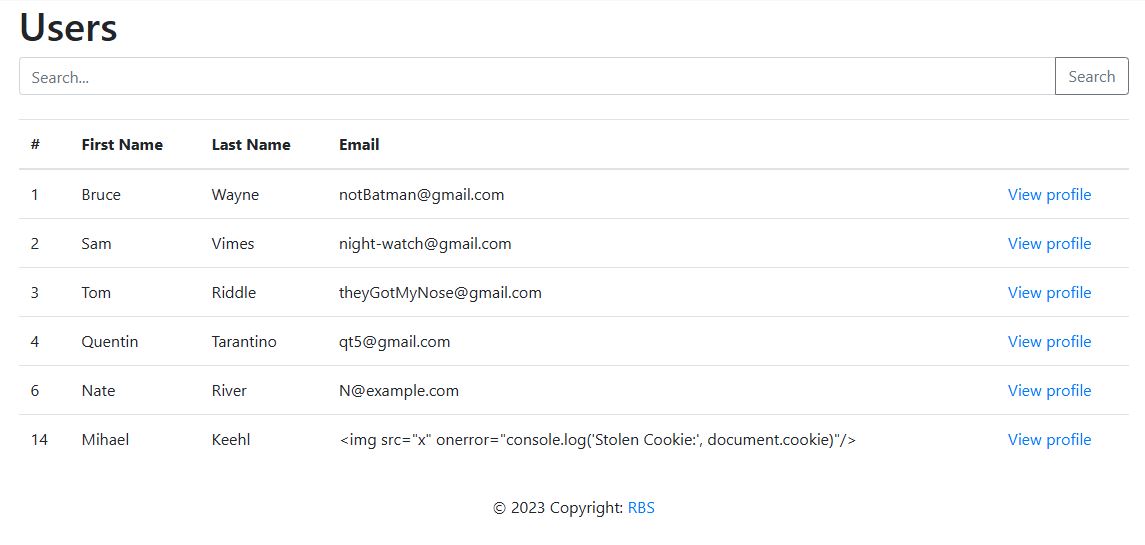


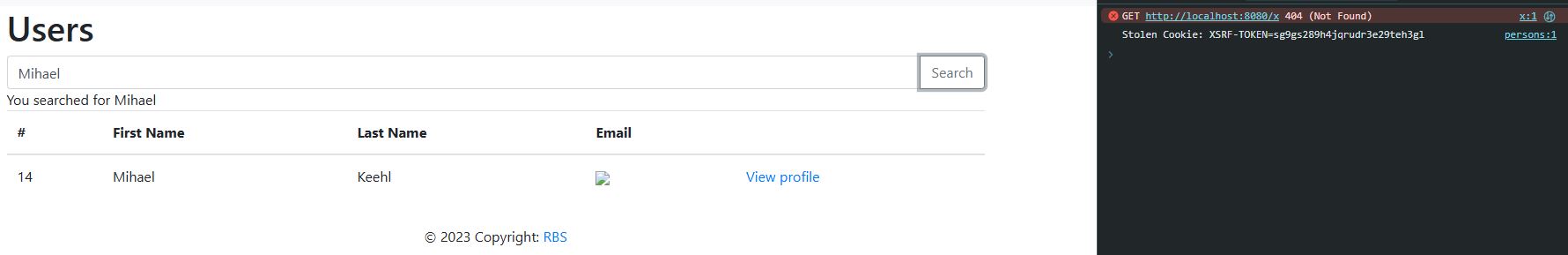
Korisnik je sada dodat.

Prelazimo na očekivano menjanje podataka:

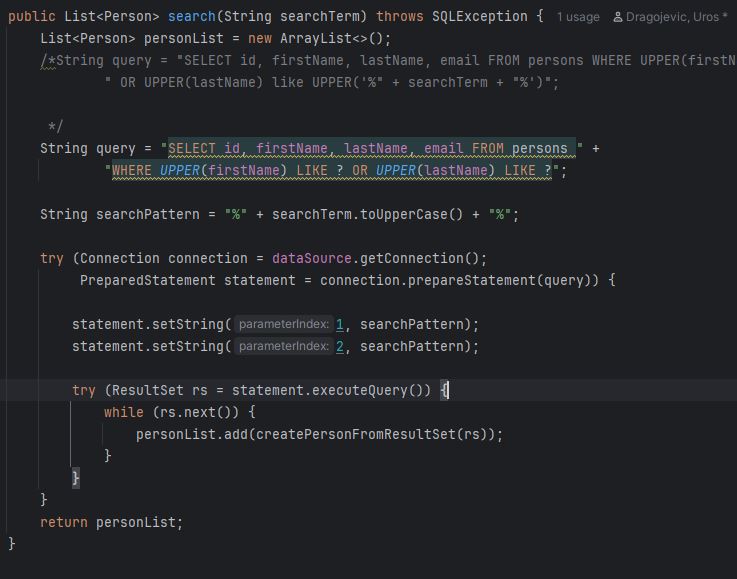
U skripti tražimo nepostojeću sliku “x” kako bismo isforsirali grešku.

Podaci su uspešno promenjeni.

Ostaje samo da aktivira skripta klikom na dugme “Search” i dobijamo Cookie ID.



## Odbrana: Za SQL injection identično, otklanjamo direktnu konkatenaciju, a za XSS menjamo .innerHtml sa .textContent





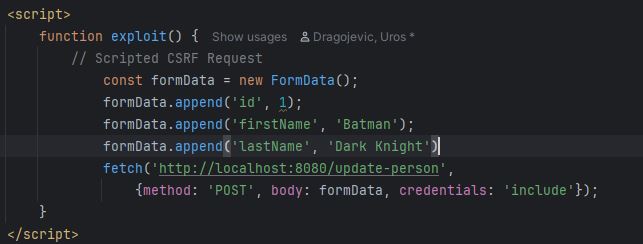
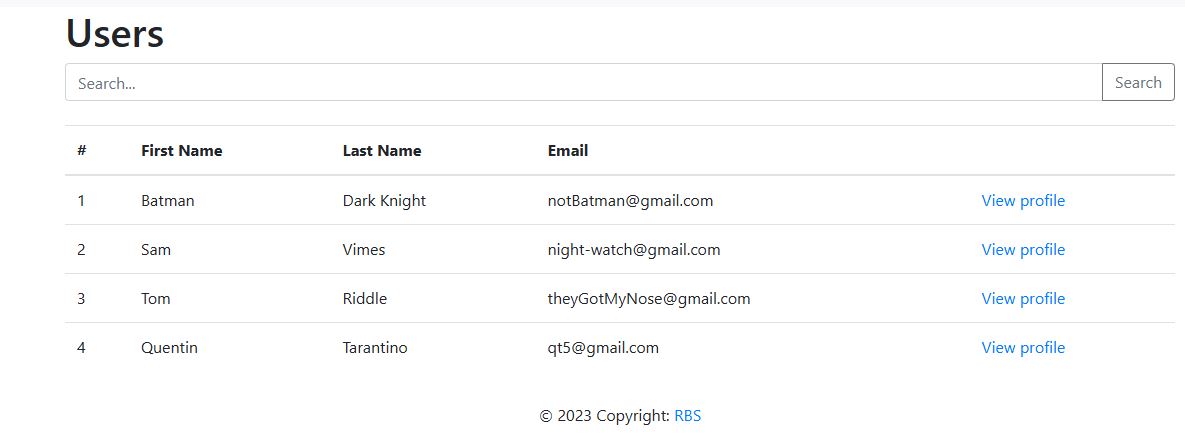
Nakon ovih promena, isprobavamo ponovo napad i može se uočiti da su ove ranjivosti otklonjene.



# Cross-site request forgery

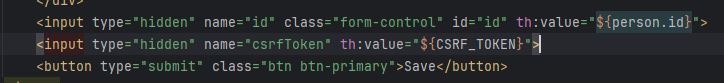
## Napad: Navodimo žrtvu da klikne link koji šalje zahtev koji menja podatke

Zahtev koji šaljemo:

  
Žrtva klikne na našu stranicu i kada ode na stranicu ”persons” podaci budu promenjeni:

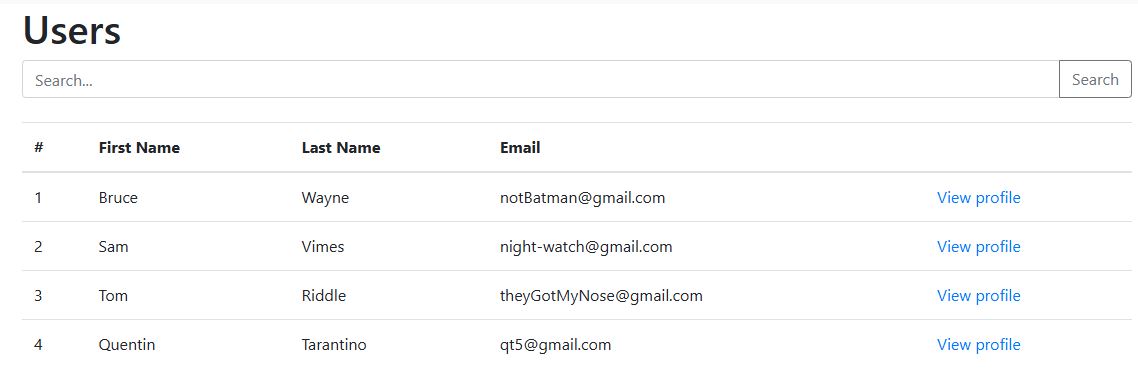
## Odbrana: Implementiramo logiku tokena







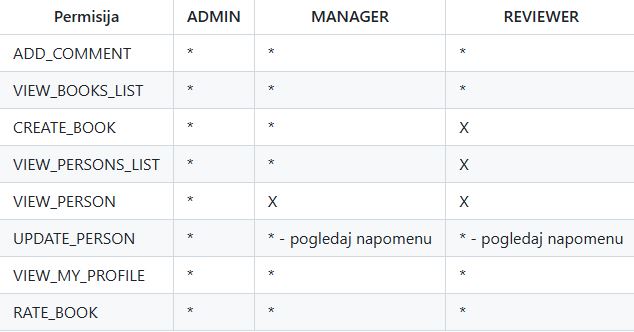
I sada nakon što pokušamo napad, ništa se ne dešava



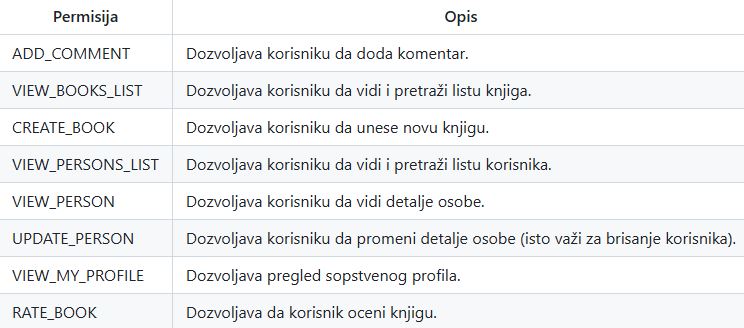
# Implementacija autorizacije

Rađeno pomoću Thymeleaf-a i Spring security framework-a.

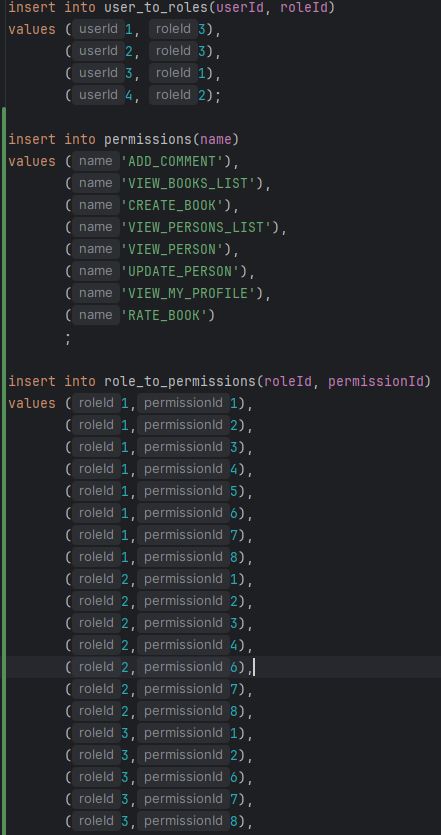
Permisije i role:



Kratko objašnjenje permisija:



Prvo ažuriramo bazu mapiranjem permisija i rola:



Potrebno je za svaku permisiju proveriti i frontend i backend (ukoliko postoji).

Za backend se dodaje anotacija: @PreAuthorize(**"hasAuthority('NAZIV\_PERMISIJE')"**)

Za frontend se html elementu dodaje atribut:

**sec:authorize="hasAuthority('NAZIV\_PERMISIJE')"**

U kodu:

