IZVEŠTAJ ZA STRUČNI KURS RBS (RAZVOJ BEZBEDNOG SOFTVERA)

Uvod

U ovom izveštaju su zabeleženi ključni momenti bezbednosnog testiranja jedne veb aplikacije.

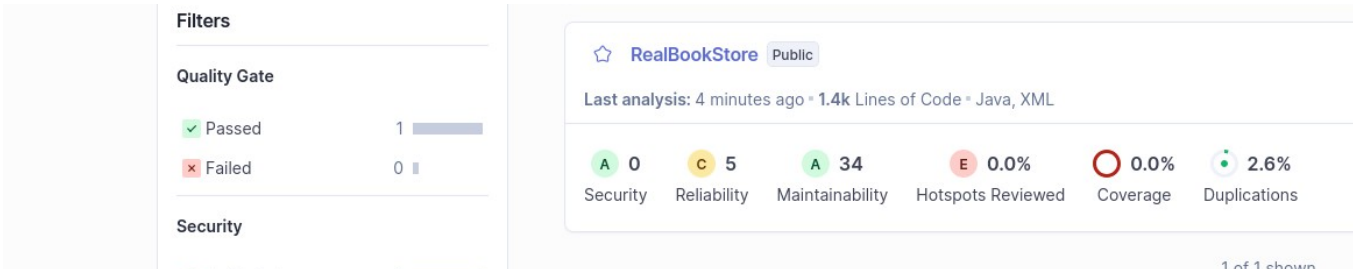
Kratak opis aplikacije

RealBookStore je veb aplikacija koja pruža mogućnost pretrage, ocenjivanja I komentarisanja knjiga.
Aplikacija omogućava sledeće:

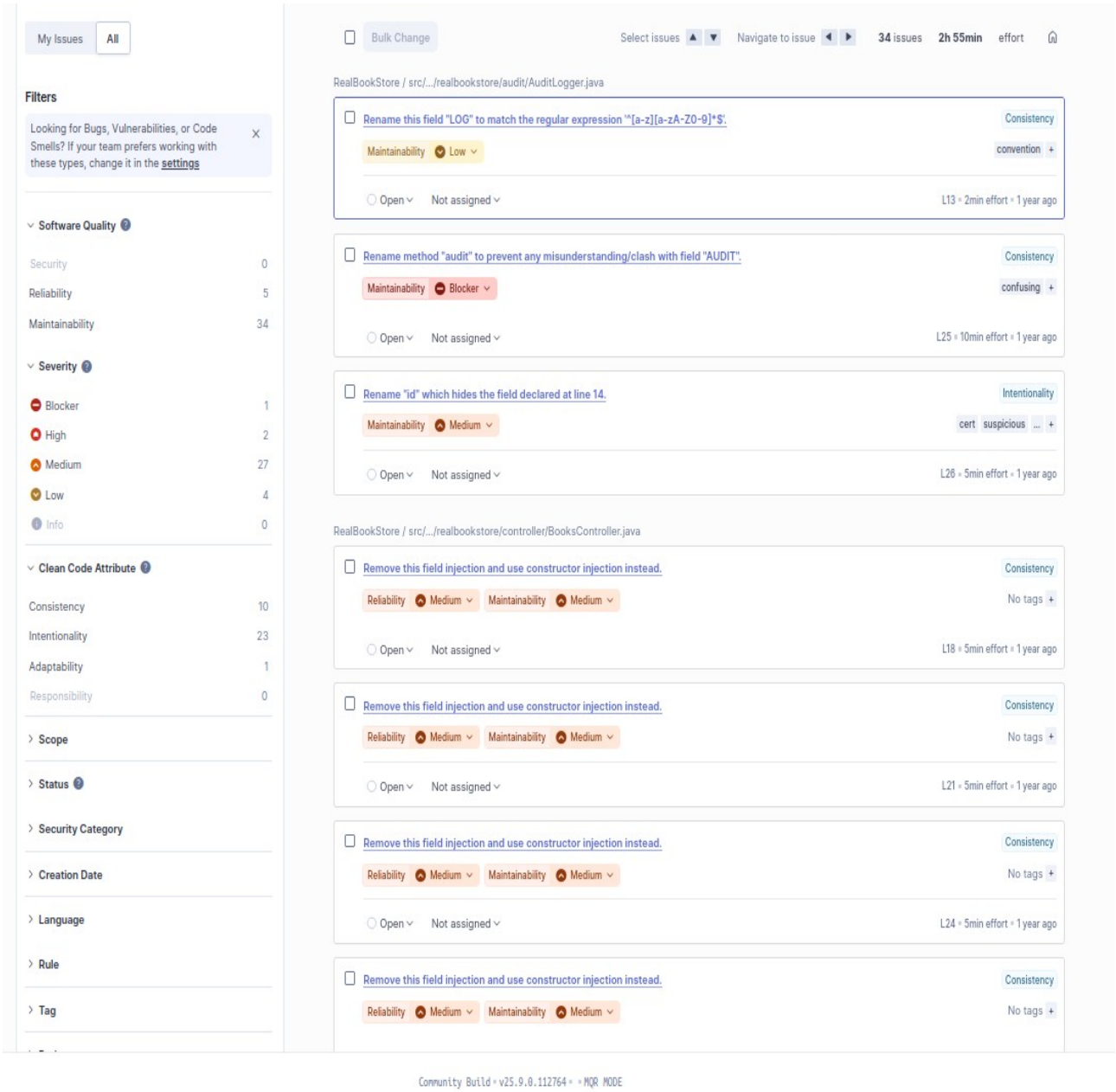
1. Pregled I pretragu knjiga
2. Dodavanje nove knjige
3. Detaljan pregled knjige kao I komentarisanje I ocenjivanje knjige
4. Pregled korisnika aplikacije
5. Detaljan pregled podataka korisnika

1.Primena alata za statičku analizu

Korišćen je SonarQube alat.
Kratak pregled broja I stepena ozbiljnosti grešaka:



Konkretne greške:



My Issues | All

Filters

Looking for Bugs, Vulnerabilities, or Code Smells? If your team prefers working with these types, change it in the settings

Software Quality

Security	0
Reliability	5
Maintainability	34

Severity

Blocker	1
High	2
Medium	27
Low	4
Info	0

Clean Code Attribute

Consistency	10
Intentionality	23
Adaptability	1
Responsibility	0

Scope

Status

Security Category

Creation Date

Language

Rule

Tag

Remove this field injection and use constructor injection instead. — Consistency
Reliability Medium · Maintainability Medium · No tags
Open · Not assigned · L30 · 5min effort · 1 year ago

Rename this method name to match the regular expression '^[a-z][a-zA-Z0-9]*$'. — Consistency
Maintainability Low · convention
Open · Not assigned · L71 · 5min effort · 1 year ago

Replace this usage of 'Stream.collect(Collectors.toList())' with 'Stream.toList()' and ensure that the list is unmodified. — Intentionality
Maintainability Medium · java16
Open · Not assigned · L85 · 5min effort · 1 year ago

RealBookStore / src/.../realbookstore/controller/CommentController.java

Remove this unused "LOG" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L15 · 5min effort · 1 year ago

RealBookStore / src/.../realbookstore/controller/PersonsController.java

Remove this unused "LOG" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L22 · 5min effort · 1 year ago

Remove this unused "auditLogger" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L23 · 5min effort · 1 year ago

Define a constant instead of duplicating this literal "person" 4 times. — Adaptability
Maintainability High · design
Open · Not assigned · L35 · 10min effort · 1 year ago

Community Build · v25.9.0.112764 · · MQR MODE

My Issues | All

Filters

Looking for Bugs, Vulnerabilities, or Code Smells? If your team prefers working with these types, change it in the settings

Software Quality

Security	0
Reliability	5
Maintainability	34

Severity

Blocker	1
High	2
Medium	27
Low	4
Info	0

Clean Code Attribute

Consistency	10
Intentionality	23
Adaptability	1
Responsibility	0

Scope

Status

Security Category

Creation Date

Language

Rule

Tag

RealBookStore / src/.../realbookstore/controller/RatingsController.java

Remove this unused "LOG" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L13 · 5min effort · 1 year ago

RealBookStore / src/.../realbookstore/domain/Comment.java

Rename field "comment" — Consistency
Maintainability Medium · brain-overload
Open · Not assigned · L6 · 10min effort · 1 year ago

RealBookStore / src/.../realbookstore/domain/Rating.java

Rename field "rating" — Consistency
Maintainability Medium · brain-overload
Open · Not assigned · L7 · 10min effort · 1 year ago

RealBookStore / src/.../realbookstore/repository/BookRepository.java

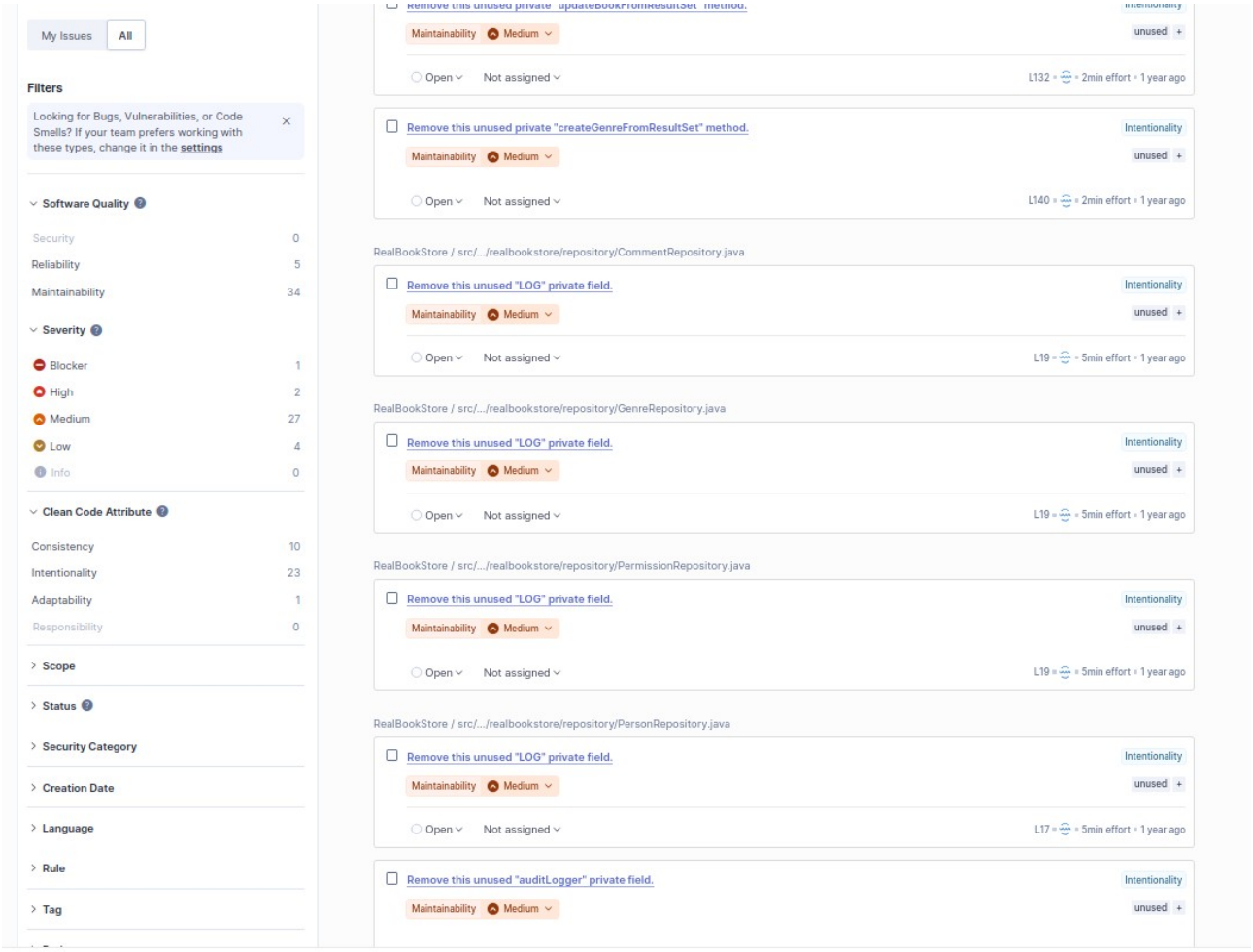
Remove this unused "LOG" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L19 · 5min effort · 1 year ago

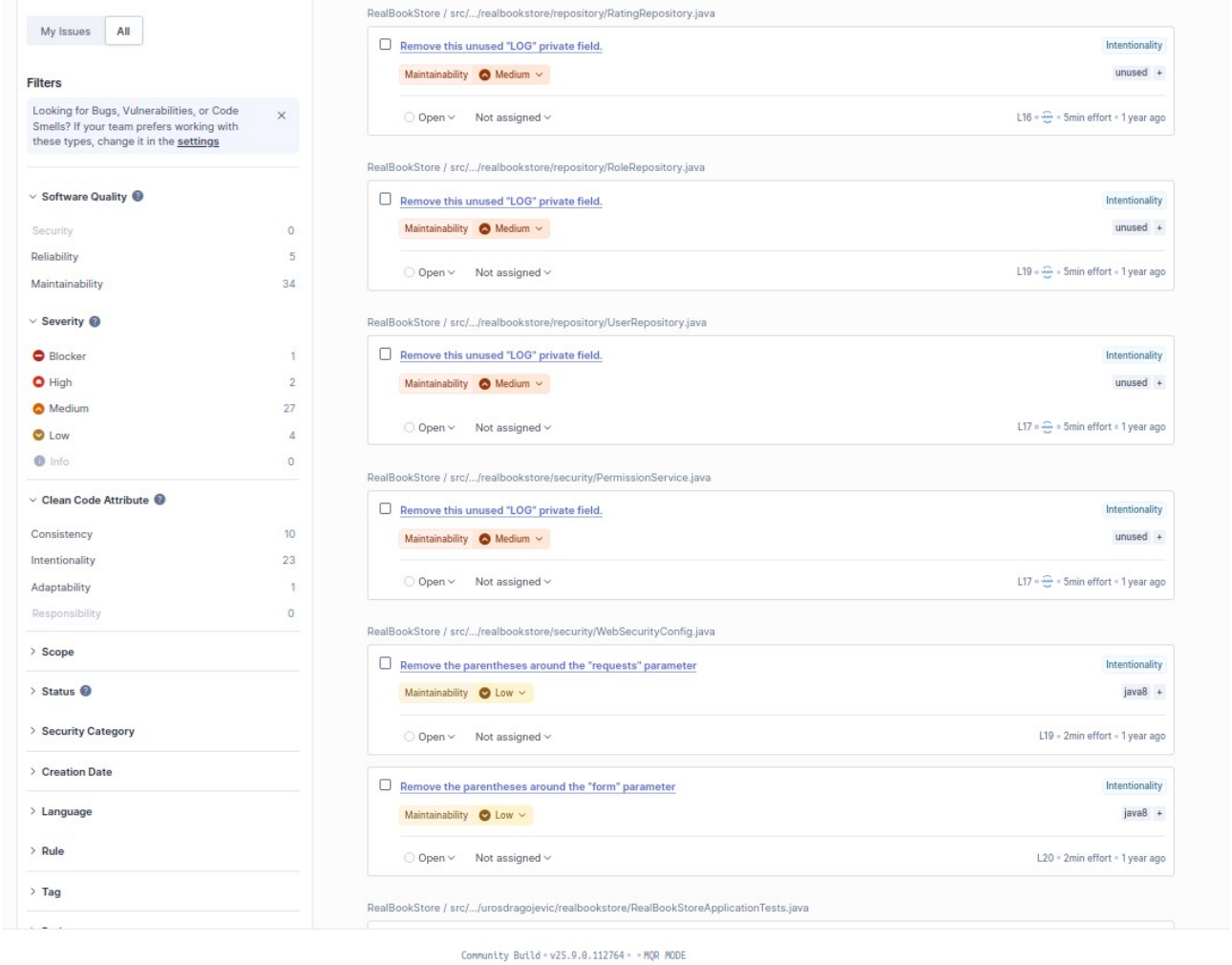
Remove this unused "auditLogger" private field. — Intentionality
Maintainability Medium · unused
Open · Not assigned · L20 · 5min effort · 1 year ago

Use "addBatch" and "executeBatch" to execute multiple SQL statements in a single call. — Intentionality
Maintainability Medium · sustainability · sql
Open · Not assigned · L95 · 5min effort · 1 year ago

Remove this unused private "updateBookFromResultSet" method. — Intentionality
Maintainability Medium · unused

Community Build · v25.9.0.112764 · · MQR MODE


My Issues
All
Filters
Looking for Bugs, Vulnerabilities, or Code Smells? If your team prefers working with these types, change it in the settings
Software Quality
Security 0
Reliability 5
Maintainability 34
Severity
Blocker 1
High 2
Medium 27
Low 4
Info 0
Clean Code Attribute
Consistency 10
Intentionality 23
Adaptability 1
Responsibility 0
Scope
Status
Security Category
Creation Date
Language
Rule
Tag
Maintainability Medium
unused +
Open
Not assigned
L132 2min effort 1 year ago
Remove this unused private "createGenreFromResultSet" method.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L140 2min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/CommentRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L19 5min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/GenreRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L19 5min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/PermissionRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L19 5min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/PersonRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L17 5min effort 1 year ago
Remove this unused "auditLogger" private field.
Intentionality
Maintainability Medium
unused +


My Issues
All
Filters
Looking for Bugs, Vulnerabilities, or Code Smells? If your team prefers working with these types, change it in the settings
Software Quality
Security 0
Reliability 5
Maintainability 34
Severity
Blocker 1
High 2
Medium 27
Low 4
Info 0
Clean Code Attribute
Consistency 10
Intentionality 23
Adaptability 1
Responsibility 0
Scope
Status
Security Category
Creation Date
Language
Rule
Tag
RealBookStore / src/.../realbookstore/repository/RatingRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L16 5min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/RoleRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L19 5min effort 1 year ago
RealBookStore / src/.../realbookstore/repository/UserRepository.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L17 5min effort 1 year ago
RealBookStore / src/.../realbookstore/security/PermissionService.java
Remove this unused "LOG" private field.
Intentionality
Maintainability Medium
unused +
Open
Not assigned
L17 5min effort 1 year ago
RealBookStore / src/.../realbookstore/security/WebSecurityConfig.java
Remove the parentheses around the "requests" parameter
Intentionality
Maintainability Low
java8 +
Open
Not assigned
L19 2min effort 1 year ago
Remove the parentheses around the "form" parameter
Intentionality
Maintainability Low
java8 +
Open
Not assigned
L20 2min effort 1 year ago
RealBookStore / src/.../urosdragojevic/realbookstore/RealBookStoreApplicationTests.java

Većina problema koji su se pojavili su greške koje utiču na održivost koda, što znači da nisu direktno sigurnosne ranjivosti.

Tipovi grešaka se kao što vidimo ponavljaju, pa nema potrebe analizirati svaku posebno. Evo nekoliko:

1. AuditLogger.java

-Problem: Rename this field "LOG" to match the regular expression

-Tip: Maintainability (Low)

-Ocena: False Positive (samo nije ispoštovana konvencija, inače je bezbedno)

2. AuditLogger.java

-Problem: Rename method "audit" to prevent clash with field "AUDIT"

-Tip: Maintainability (Blocker)

-Ocena: True Positive (nezgodna, plodno tle za bug)

3.BooksController.java

-Problem: Remove this field injection and use constructor injection instead

-Tip: Reliability & Maintainability (Medium)

-Ocena: True Positive (problemi pri testiranju mogući)

4.RatingsController.java

-Problem: Remove this unused "LOG" private field

-Tip: Maintainability (Medium)

-Ocena: True Positive (ako se ne koristi onda nije potrebna)

5.BookStore.java

-Problem: Use „addBatch“ and „executeBatch“ to execute multiple SQL statements in a single call

-Tip: Maintainability (Medium)

-Ocena: True Positive (utiče na čitljivost)

6.PersonsController.java

-Problem: Define a constant instead of duplicating this literal "person" 4 times

-Tip: Maintainability (High)

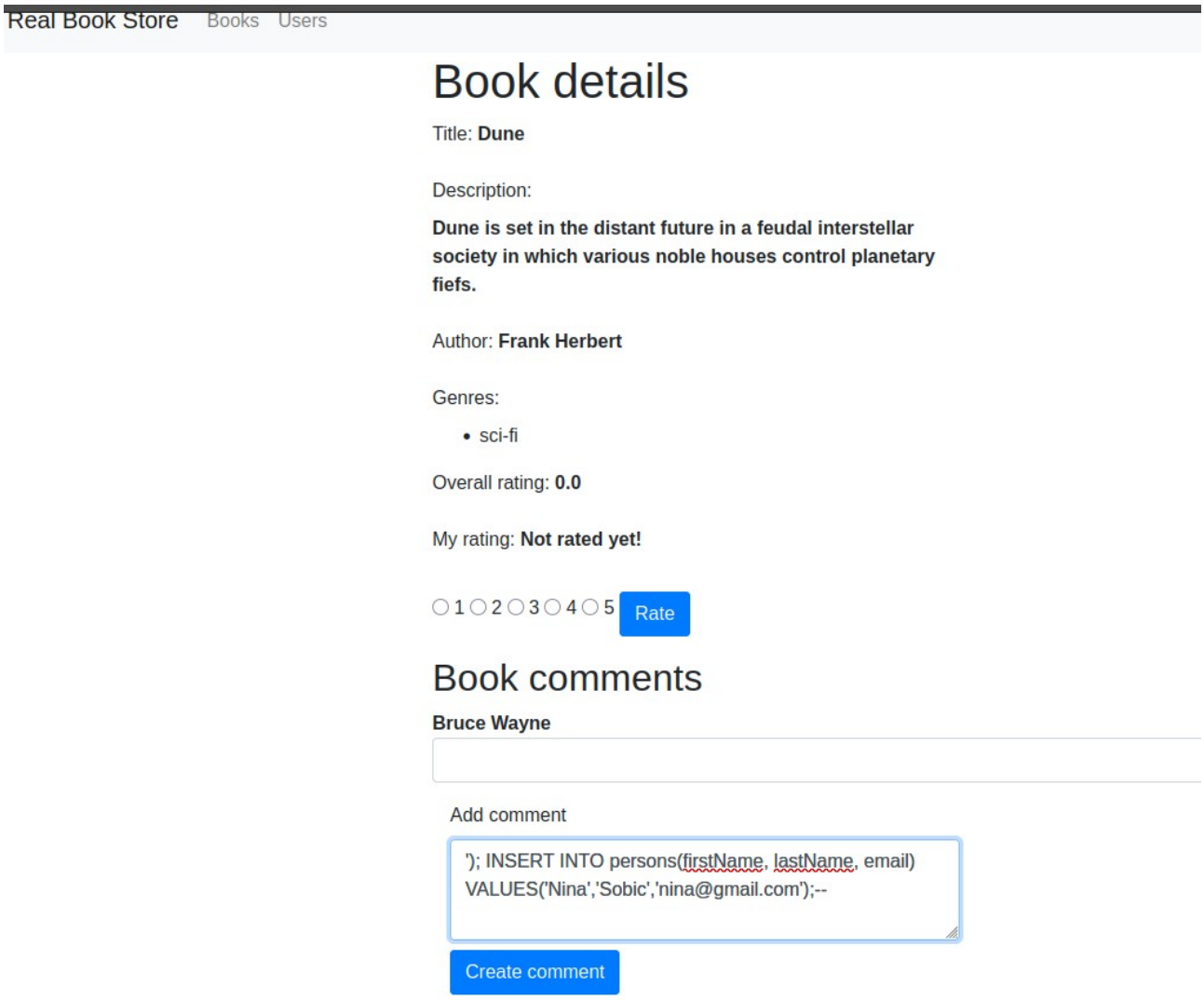
-Ocena: False Positive (utiče na čitljivost I održavanje)

2.SQL Injection i Cross-site scripting

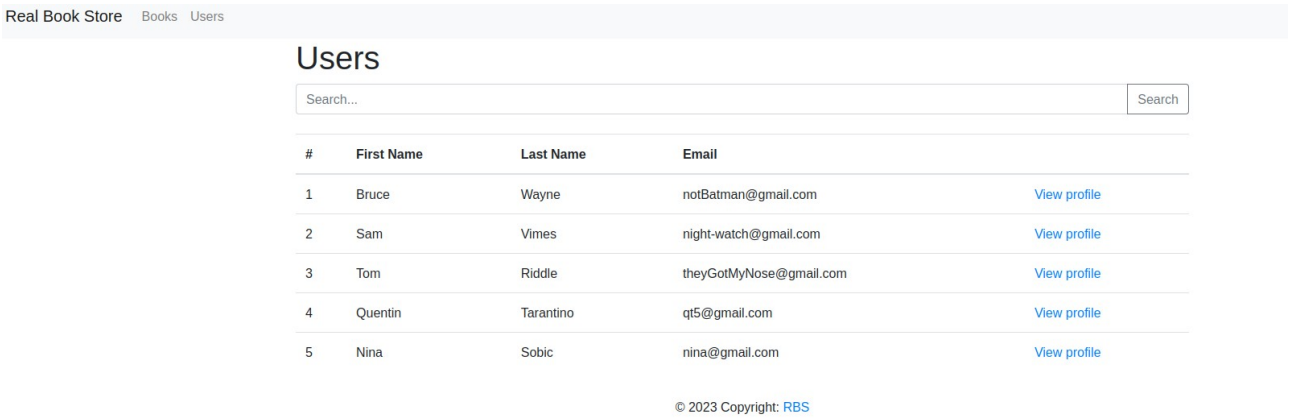
*SQL injection

-Želimo da proverimo da li je aplikacija podložna ovoj vrsti napada.

Napad:



Sledeća slika pokazuje da je **napad uspeo**:



Moguća **odbrana**: koristiti PreparedStatement umesto Statement.

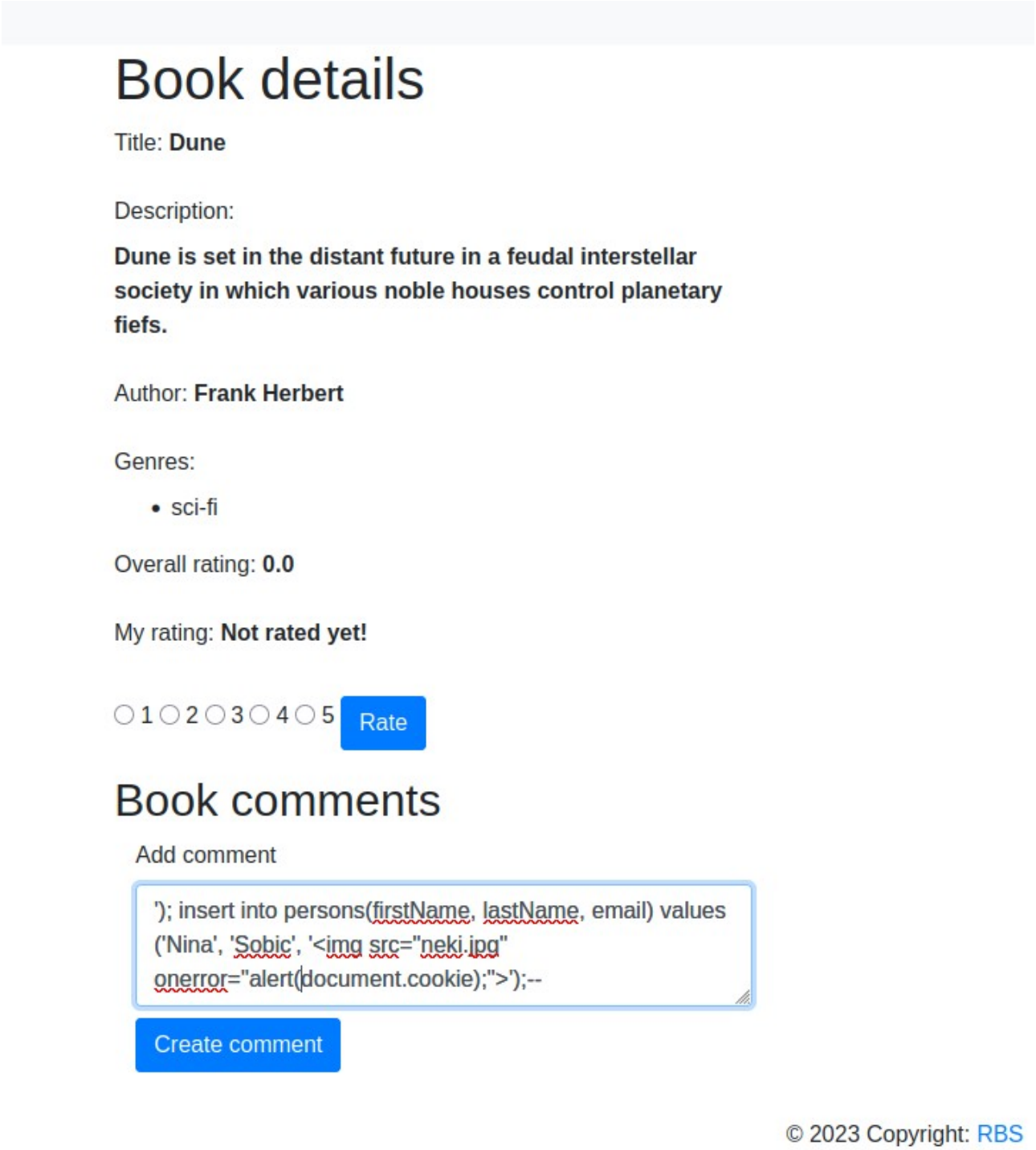
```
public void create(Comment comment) {  1 usage  new *
    String query = "insert into comments(bookId, userId, comment) values (?,?,?)";

    try (Connection connection = dataSource.getConnection();
         PreparedStatement statement = connection.prepareStatement(query);
    ) {
        statement.setInt( parameterIndex: 1, comment.getBookId());
        statement.setInt( parameterIndex: 2, comment.getUserId());
        statement.setString( parameterIndex: 3, comment.getComment());
        statement.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

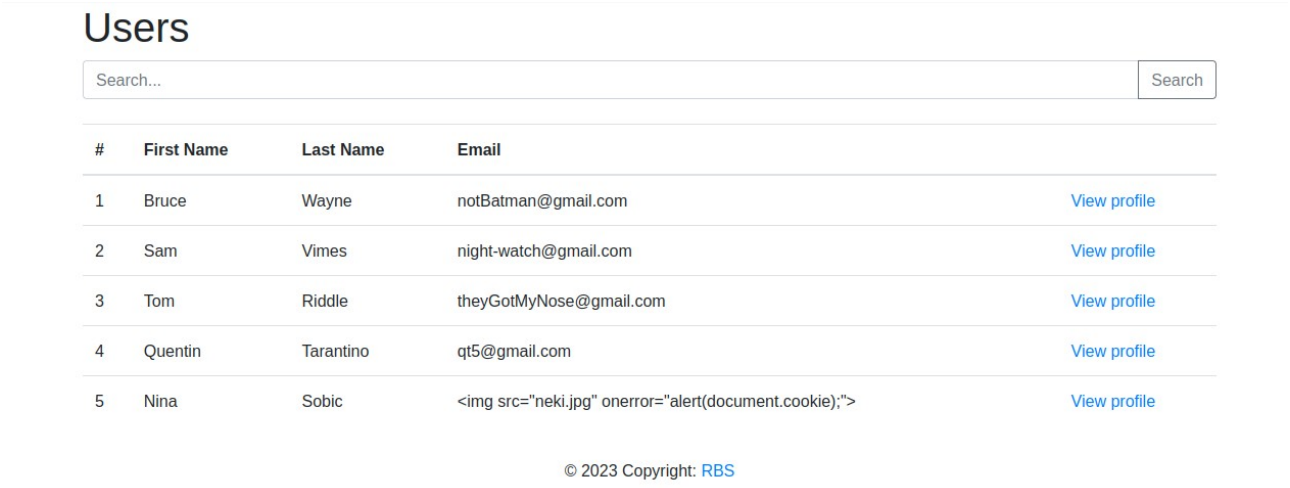
Sada, napad neće uspeti, već će se samo komentar dodati. To I želimo:

*Cross-site scripting

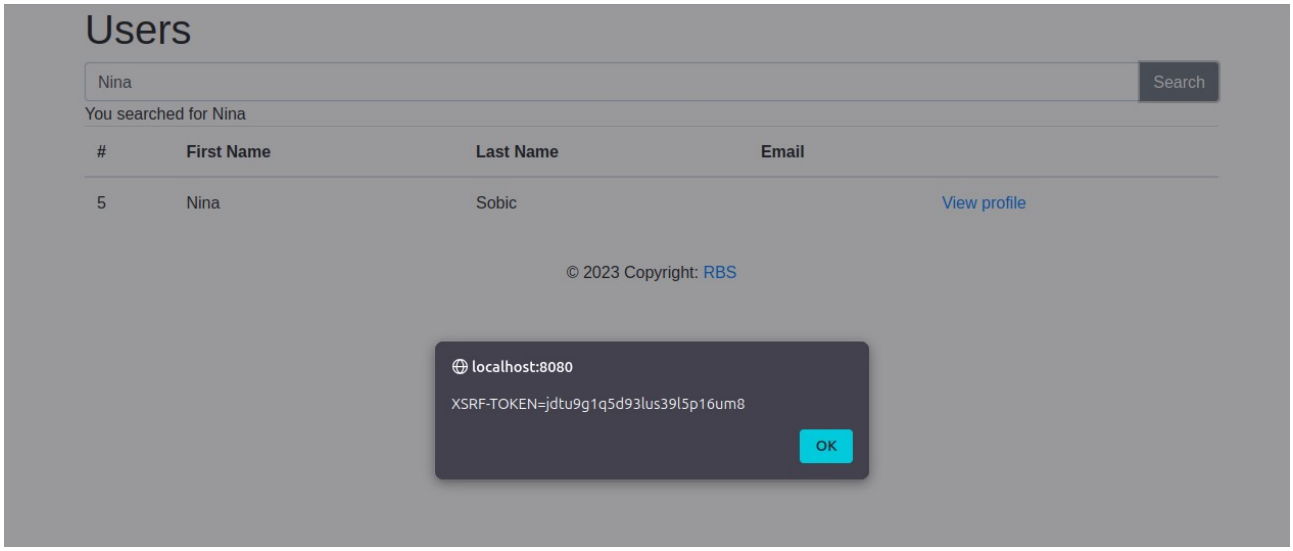
Pokušaj **napada**:



Napad uspeo, **rezultat**:



Kada pokušamo da pretražimo novog korisnika:



Odbrana je PreparedStatement umesto Statement I textContent umesto innerHTML. Na primer:

```
    persons.forEach(function(person) {
        const tableRowElement = document.createElement("tr");
        let tdElement = document.createElement("td");
        tdElement.textContent = person.id;
        tableRowElement.appendChild(tdElement);
        tdElement = document.createElement("td");
        tdElement.textContent = person.firstName;
        tableRowElement.appendChild(tdElement);
        tdElement = document.createElement("td");
        tdElement.textContent = person.lastName;
        tableRowElement.appendChild(tdElement);
        tdElement = document.createElement("td");
        tdElement.textContent = person.email;
        tableRowElement.appendChild(tdElement);
        tdElement = document.createElement("td");
        tdElement.textContent = '<a href="/persons/' + person.id + '">View profile</a>';
        tableRowElement.appendChild(tdElement);

        tableContent.appendChild(tableRowElement);
    });

    document.getElementById('searchContainer').className = '';
    document.getElementById('searchTerm').textContent = searchTerm;
});
```

Sada ako pokušamo pretragu, sve će biti u redu!

Cross-Site Request Forgery (CSRF)

Napad: Klikom na maliciozni link, pokreće se skripta koja šalje zahtev serveru i menja podatke korisnika sa id = 1, tako da je firstName Batman i lastName Dark Knight.

Funkcija:

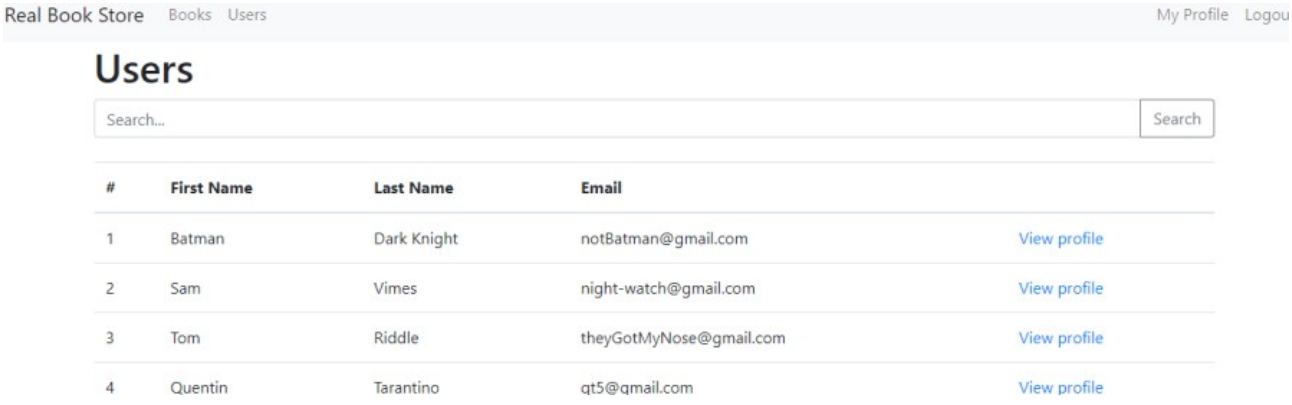
```
function exploit() {  Show usages   Dragojevic, Uros *
    // Scripted CSRF Request
    const formData = new FormData();
    formData.append('id', '1');
    formData.append('firstName', 'Batman');
    formData.append('lastName', 'Dark Knight');
    fetch('http://localhost:8080/update-person', {
        method: 'POST',
        body: formData,
        credentials: 'include'
    });
}
```

Klikom na link-sliku:

YOU WIN!

Click here!

Menjaju se podaci korisnika čiji je id=1:



Moguća odbrana: korišćenje CSRF tokena koji se čuvaju tokom sesije.

```
@GetMapping("/persons/{id}")  Dragojevic, Uros *
public String person(@PathVariable int id, Model model, HttpSession session) {
    model.addAttribute( attributeName: "CSRF_TOKEN", session.getAttribute( s: "CSRF_TOKEN"));
    model.addAttribute( attributeName: "person", personRepository.get("" + id));
    return "person";
}
```

```
@PostMapping("/update-person")  new *
public String updatePerson(Person person, HttpSession session, @RequestParam("csrfToken") String csrfToken) throws AccessDeniedException {
    String token = session.getAttribute( s: "CSRF_TOKEN").toString();
    if(!token.equals(csrfToken))
        throw new AccessDeniedException("Forbidden");

    Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
    User current = (User) authentication.getPrincipal();
    if(Integer.parseInt(person.getId()) != current.getId()) {
        throw new AccessDeniedException("You can only update your own profile");
    }

    personRepository.update(person);
    return "redirect:/persons/" + person.getId();
}

@GetMapping("/persons")  Dragojevic, Uros
```

Autorizacija

Implementirati autorizacioni model (matricu permisija) u bazi podataka.

```
insert into permissions(id, name)
values (1, 'ADD_COMMENT'),
       (2, 'VIEW_BOOKS_LIST'),
       (3, 'CREATE_BOOK'),
       (4, 'VIEW_PERSONS_LIST'),
       (5, 'VIEW_PERSON'),
       (6, 'UPDATE_PERSON'),
       (7, 'VIEW_MY_PROFILE'),
       (8, 'RATE_BOOK');
```

```
insert into role_to_permissions(roleId, permissionId)
values (1, 1),
       (1, 2),
       (1, 3),
       (1, 4),
       (1, 5),
       (1, 6),
       (1, 7),
       (1, 8),
       (2, 1),
       (2, 2),
       (2, 3),
       (2, 4),
       (2, 6),
       (2, 7),
       (2, 8),
       (3, 1),
       (3, 2),
       (3, 6),
       (3, 7),
       (3, 8);
```

Korišćeni su Spring Security i Thymeleaf koncepti.

DevOps

Radi praćenje promena i grešaka:
-Uvedena obrada i logovanje svih izuzetaka u aplikaciji. Dodati logovi koji bi bili korisni u analizi u slučaju napada.
-Uveden auditing aplikaciji.
Primer koda:

```
@Controller  Dragojevic, Uros +1
public class CommentController {
    private static final Logger LOG = LoggerFactory.getLogger(CommentController.class);  no usages
    private static final AuditLogger auditLogger = AuditLogger.getAuditLogger(CommentController.class);  1 usage
    private CommentRepository commentRepository;  2 usages

    public CommentController(CommentRepository commentRepository) { this.commentRepository = commentRepository; }

    @PostMapping(value = "/comments")  Dragojevic, Uros +1
    @PreAuthorize("hasAuthority('ADD_COMMENT')")
    public String createComment(@ModelAttribute Comment comment, Authentication authentication) {
        User user = (User) authentication.getPrincipal();
        comment.setUserId(user.getId());
        commentRepository.create(comment);

        auditLogger.audit( description: "Added comment "+ comment.toString());

        return "redirect:/books/" + comment.getBookId();
    }
}
```

Zamenjeni su svi printStackTrace pozivi logovanjem radi lakšeg praćenja u slučaju greške.

```
public void delete(int bookId) {  no usages  Dragojevic, Uros +1
    String query = "DELETE FROM books WHERE id = " + bookId;
    String query2 = "DELETE FROM ratings WHERE bookId = " + bookId;
    String query3 = "DELETE FROM comments WHERE bookId = " + bookId;
    String query4 = "DELETE FROM books_to_genres WHERE bookId = " + bookId;
    try (Connection connection = dataSource.getConnection();
         Statement statement = connection.createStatement();
    ) {
        statement.executeUpdate(query);
        statement.executeUpdate(query2);
        statement.executeUpdate(query3);
        statement.executeUpdate(query4);
    } catch (SQLException e) {
        LOG.error("Deleting book by ID failed.", e);
    }
}
```