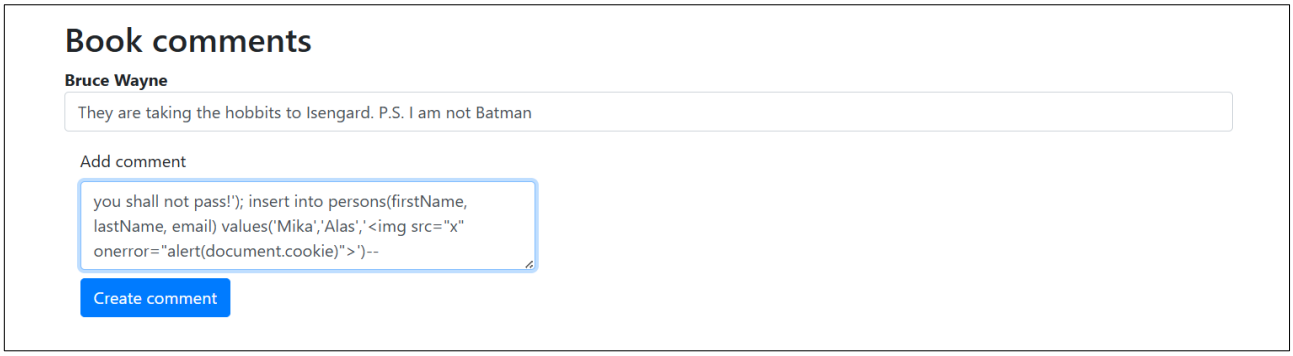
Razvoj bezbednog softvera

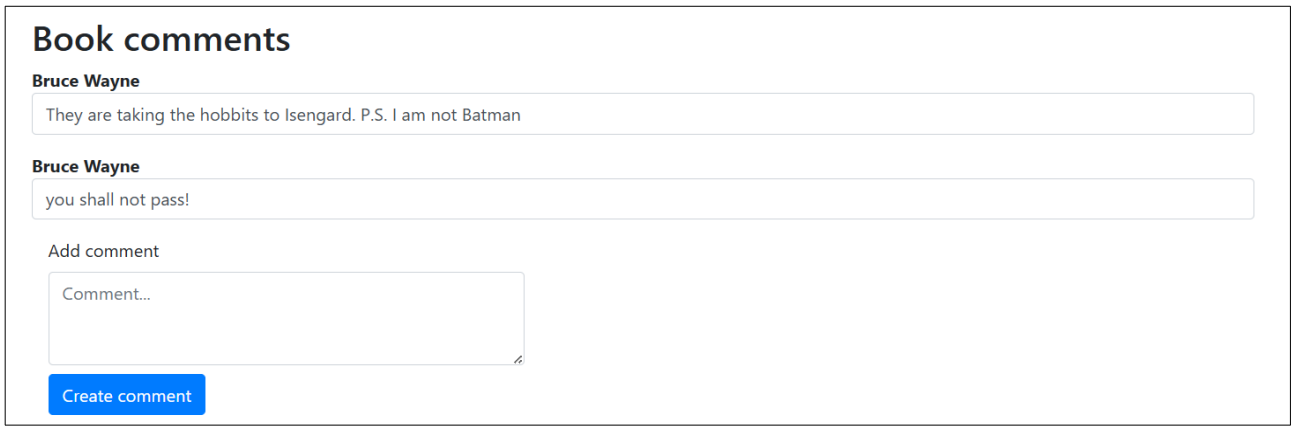
Izveštaj o izradi projekta

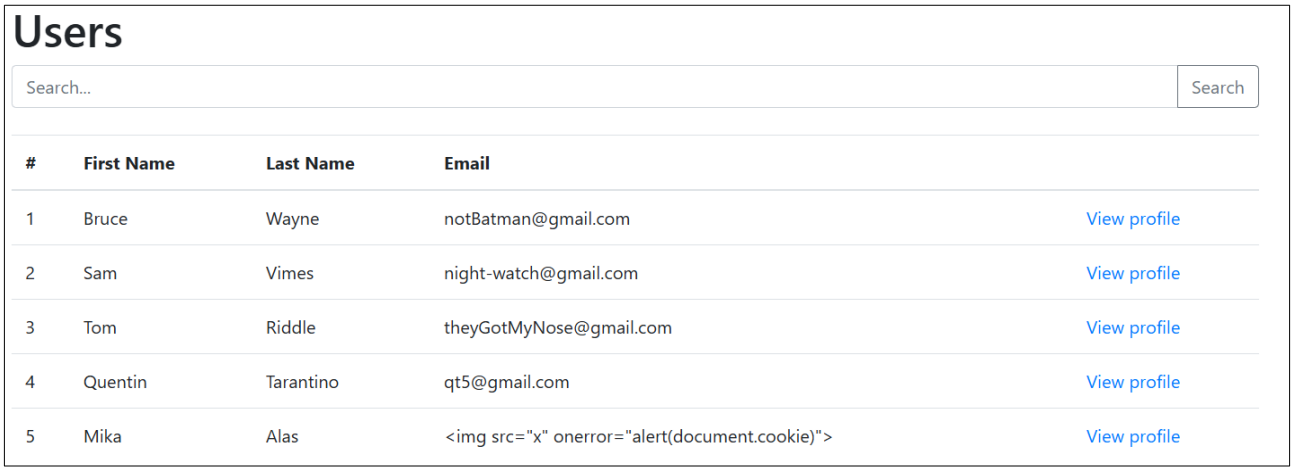
Marko Bekonja, 432/2019

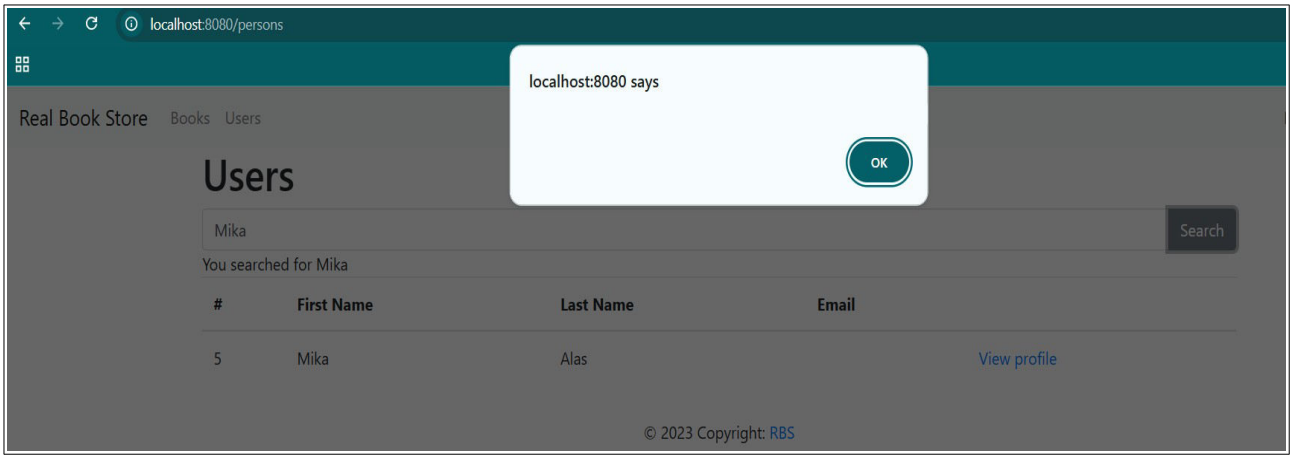
SQL Injection i Cross-site scripting

Napad:









Odbrana:

```java
public void create(Comment comment) {  1 usage  Dragojevic, Uros *
    String query = "INSERT INTO comments(bookId, userId, comment) VALUES (?, ?, ?)";

    try (Connection connection = dataSource.getConnection();

        PreparedStatement preparedStatement = connection.prepareStatement(query)) {

            preparedStatement.setInt( parameterIndex: 1, comment.getBookId());
            preparedStatement.setInt( parameterIndex: 2, comment.getUserId());
            preparedStatement.setString( parameterIndex: 3, comment.getComment());

            preparedStatement.executeUpdate();

        } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```java
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");
    const link = document.createElement("a");
    link.href = "/persons/" + person.id;
    link.textContent = "View profile";
    tdElement.appendChild(link);
    tableRowElement.appendChild(tdElement);

    tableContent.appendChild(tableRowElement);
});
```

Komentar: S obzirom da je session cookie tipa httpOnly, u mom pretraživaču nije mogao da se prikaže ali je svakako proizvedeno neočekivano ponašanje kao posledica XSS napada.

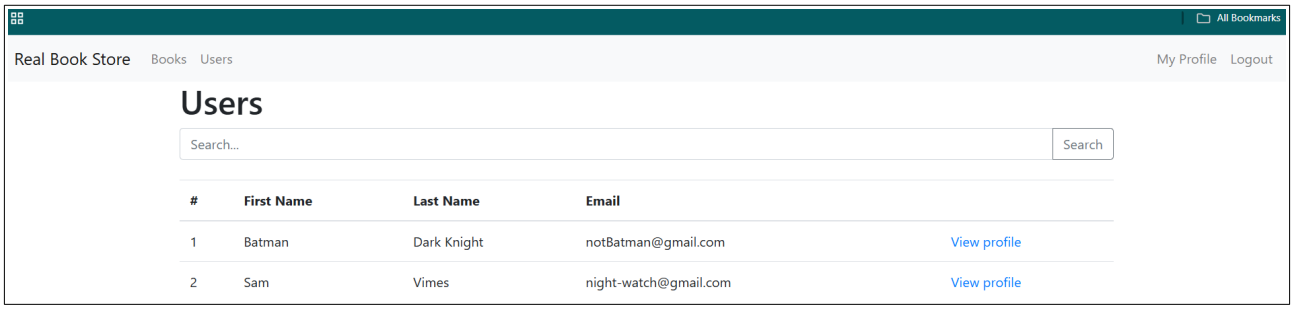
Cross-site request forgery

Napad:

```javascript
function exploit() {  Show usages  Dragojevic, Uros *

    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'});

}
```



Odbrana:

```java
@GetMapping("/persons/{id}")  Dragojevic, Uros *
public String person(@PathVariable int id, Model model, HttpSession session) {

    String csrf = session.getAttribute( s: "CSRF_TOKEN").toString();
    model.addAttribute( attributeName: "CSRF_TOKEN", session.getAttribute( s: "CSRF_TOKEN"));
    model.addAttribute( attributeName: "person", personRepository.get("" + id));
    return "person";
}
```

```java
@PostMapping("/update-person")  Dragojevic, Uros *
public String updatePerson(Person person, HttpSession session, @RequestParam("csrfToken") String csrfToken) throws AccessDeniedException {
    String csrf = session.getAttribute( s: "CSRF_TOKEN").toString();
    if (!csrf.equals(csrfToken)) {
        throw new AccessDeniedException("Forbidden");
    }
    personRepository.update(person);
    return "redirect:/persons/" + person.getId();
}
```

```html
<input type="hidden" name="id" class="form-control" id="id" th:value="${person.id}">
<input type="hidden" name="csrfToken" th:value="${CSRF_TOKEN}">
<button type="submit" class="btn btn-primary">Save</button>
```

Napomena: Prilikom izrade poslednjeg zadatka koji se tiče rukovanja izuzecima, pisanja logova i implementacije auditinga, komentare sam pisao na srpskom jeziku. Svestan sam da je sve pisano na engleskom jeziku i da to što sam uradio nije dobra praksa ali sam taj propust primetio tek kada sam završio izradu zadatka. Nadam se da se neće uzeti za zlo. Takođe, prilikom izrade projekta, primetio da sam na još nekoliko mesta mogućnost da se implementira zaštita od SQL Injection napada, upotrebom PreparedStatement-a.