**Identifying and Exploiting SQL Injection Vulnerabilities in 'To Do List' Web Application**

To exploit the SQL injection vulnerability in the "To Do List" web application, we made modifications to the **remove.jsp** page and **Main.java**. After running the application with the added SQL Injection code, we input random data into the table and observed that when pressing the **“Delete”** button, items were removed without confirmation. This confirms the successful execution of the SQL Injection.

Breaking down how the vulnerability was created and exploited:

1. **Injection Attempt:** SQL code was injected into **Main.java** and **remove.jsp**, attempting to exploit the vulnerability by injecting malicious SQL statements.
2. **Unintended Data Deletion:** Upon interaction with the web page, it was observed that pressing the **“Delete”** button resulted in the permanent removal of added items from the table without confirmation, indicating successful execution of the injected SQL code.
3. **No Confirmation:** The absence of confirmation before data deletion suggests that the injected SQL code bypassed validation or authorization checks and directly manipulated the database.

**Conclusion:**

These points confirm the successful SQL injection, leading to unintended data deletion. Addressing SQL injection vulnerabilities promptly is essential to prevent unauthorized access and data manipulation in the application.