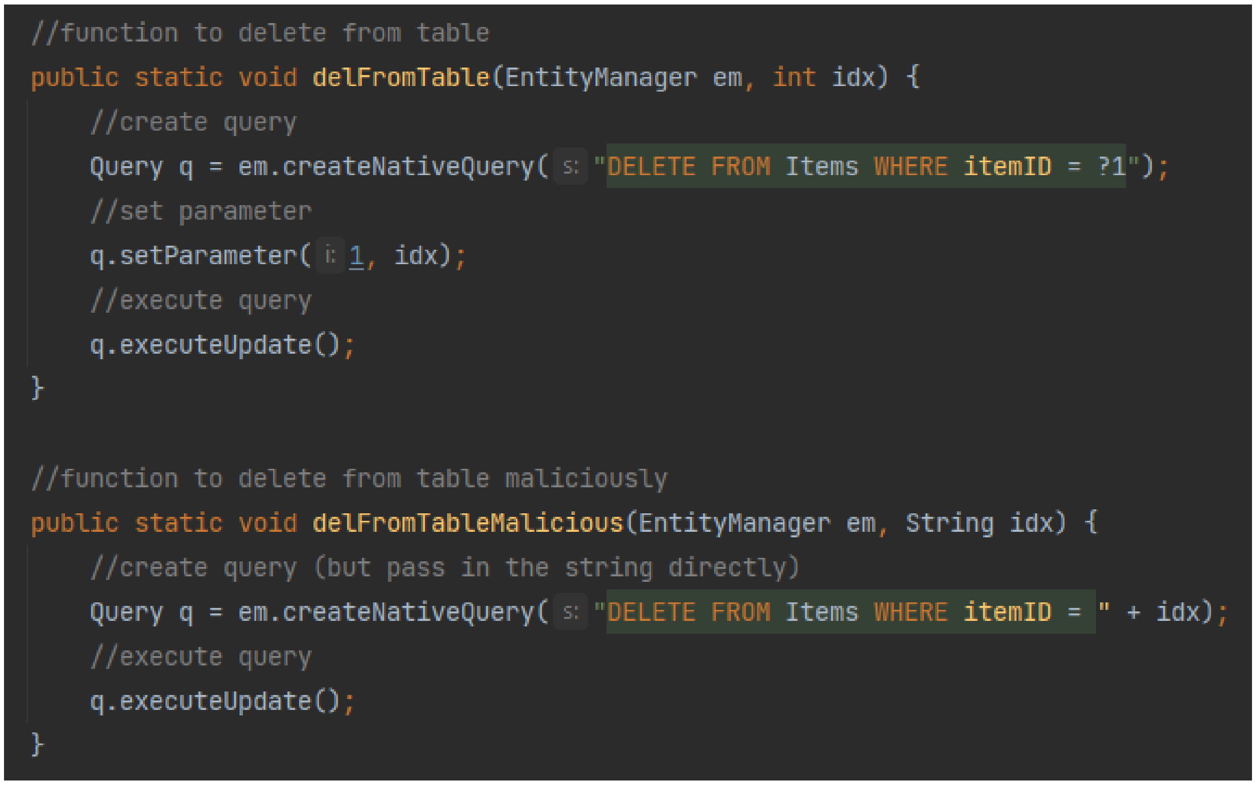
Tristin Young
CEN 4025
Fall 2023
Module 11

To create an SQL injection, multiple portions of the code needed to be changed. Previously, I had one servlet function and used a switch variable to choose which method to invoke. To introduce an SQL injection, I needed the user to be able to input the index while choosing an item to delete (previously, all items had a corresponding delete button and a logically set parameter 'itemId' to ensure minimal errors). I removed the 'delete' button from each to-do list item, and I created a new button that redirected the user to a page with a form to enter in the itemId to delete. Additionally, the variable type accepted in the itemId field needed to be a string to inflict damage, not an integer. Finally, instead of using parameters, which parse out special characters and prevent SQL injection attacks, I passed in the user set parameters directly. These changes all combined make my program vulnerable to injections, specifically on the delete.jsp page.

Here are the two methods in question. The first is a safe, injection-free implementation that will not allow an attack. The second, however, take the user input as a literal string, thus omitting any escaping of characters and leaving the program open to injection attacks. Also note how the first method has a tighter variable scope, requiring an integer as opposed to the second, which accepts a string.

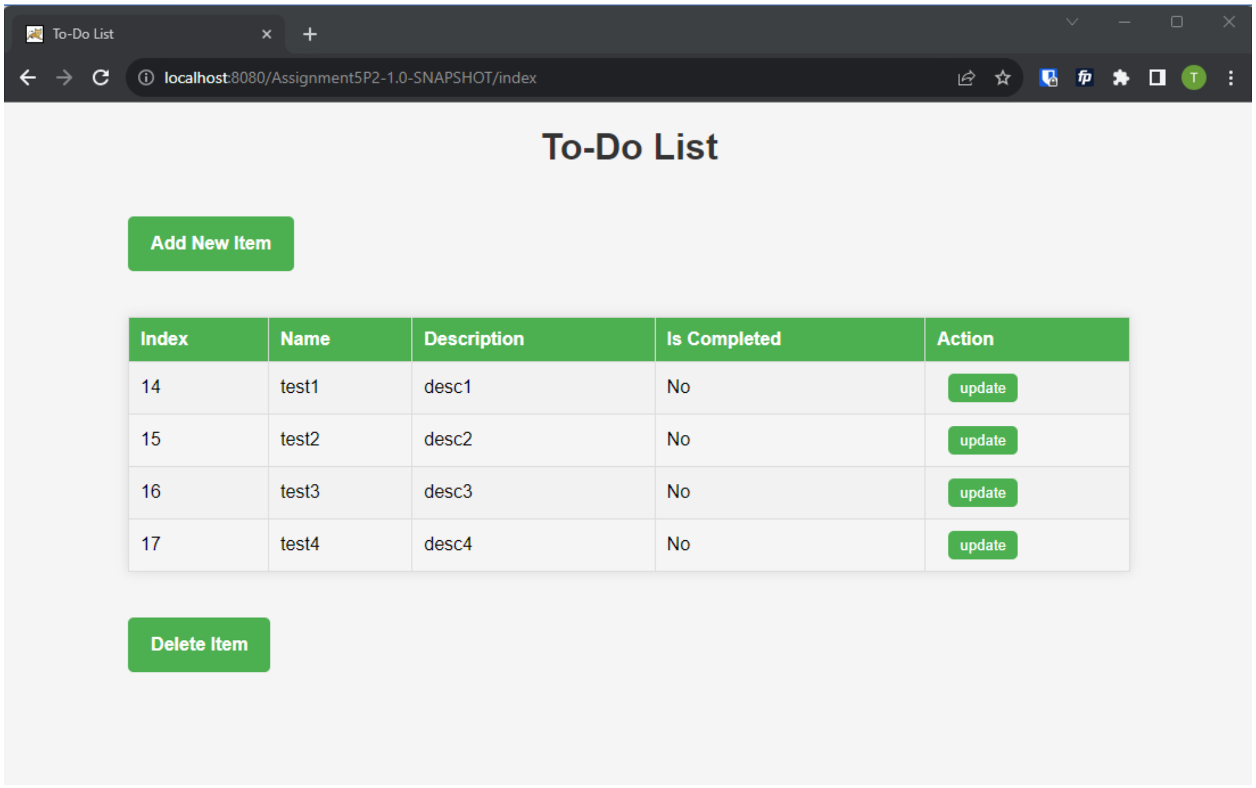


```java
//function to delete from table
public static void delFromTable(EntityManager em, int idx) {
    //create query
    Query q = em.createNativeQuery( s: "DELETE FROM Items WHERE itemID = ?1");
    //set parameter
    q.setParameter( i: 1, idx);
    //execute query
    q.executeUpdate();
}

//function to delete from table maliciously
public static void delFromTableMalicious(EntityManager em, String idx) {
    //create query (but pass in the string directly)
    Query q = em.createNativeQuery( s: "DELETE FROM Items WHERE itemID = " + idx);
    //execute query
    q.executeUpdate();
}
```

To execute an SQL injection attack, we simply insert SQL code into the input field instead of the desired integer. Specifically, we enter '0 OR 1=1'. Because we are not omitting special characters (the equals sign) and because we are passing in a string literal as opposed to an integer, the input will act as SQL code and execute on the database. This little code snippet would delete all entries in a table, which could be very detrimental depending on the data.

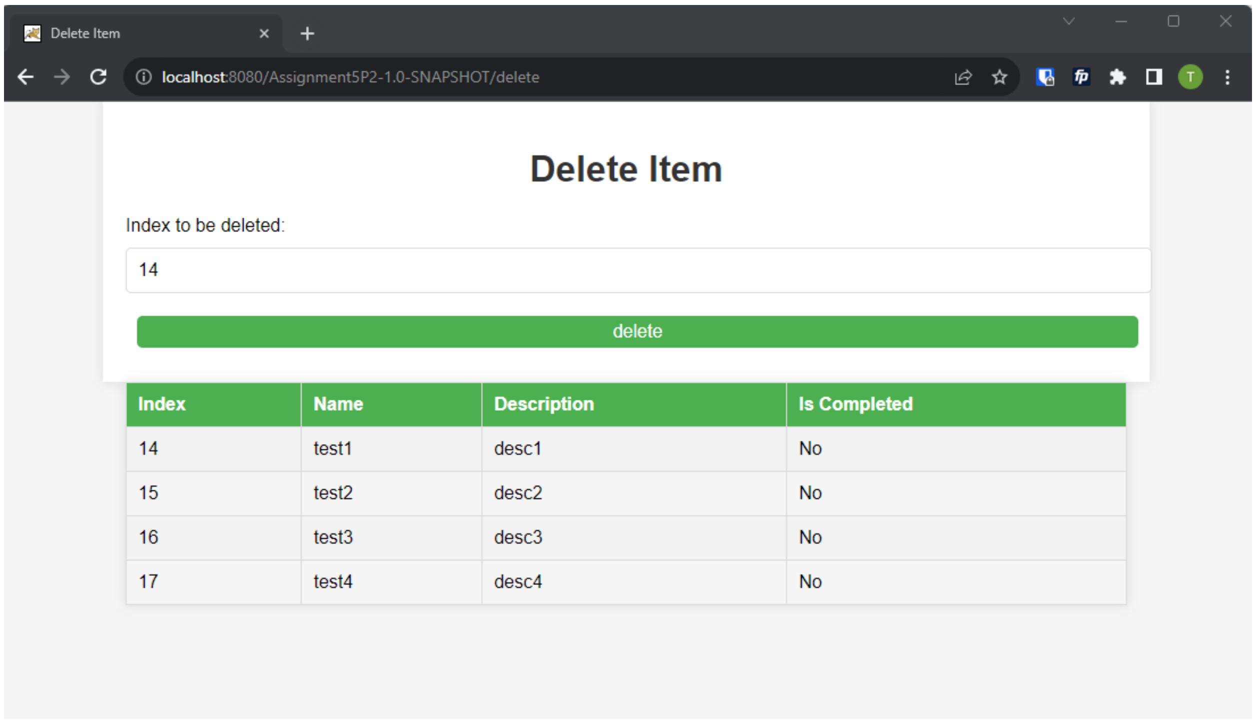
For example, this is the database before performing the attack.



To-Do List

Index	Name	Description	Is Completed	Action
14	test1	desc1	No	update
15	test2	desc2	No	update
16	test3	desc3	No	update
17	test4	desc4	No	update

We see that inserting a normal itemId into the delete input field works as expected, as seen below with the item that has index 14.


Delete Item

localhost:8080/Assignment5P2-1.0-SNAPSHOT/delete

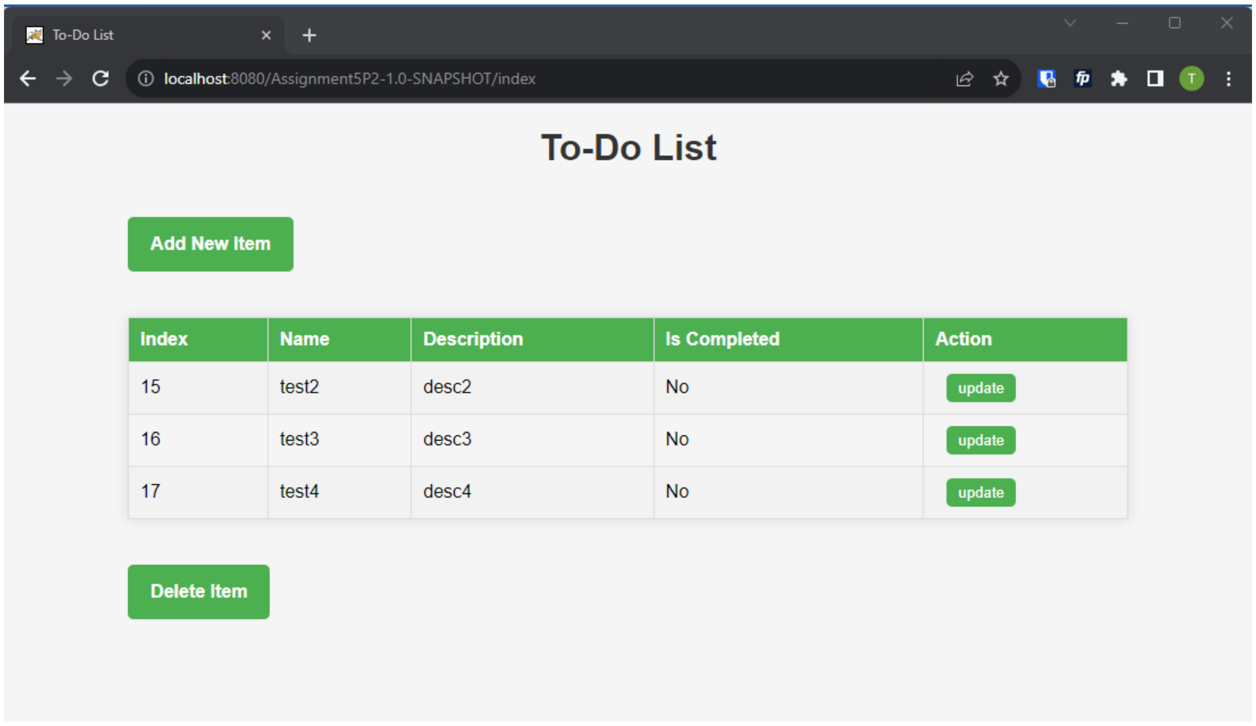
Delete Item

Index to be deleted:

14

delete

Index	Name	Description	Is Completed
14	test1	desc1	No
15	test2	desc2	No
16	test3	desc3	No
17	test4	desc4	No


To-Do List

localhost:8080/Assignment5P2-1.0-SNAPSHOT/index

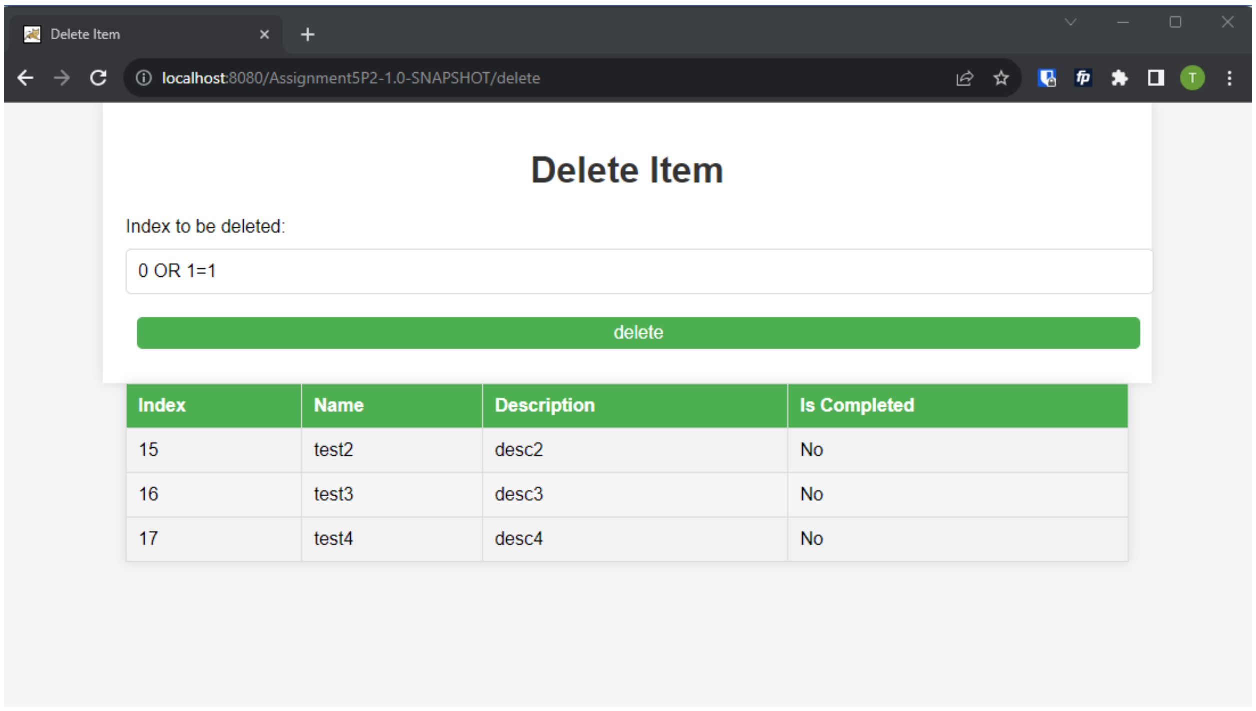
To-Do List

Add New Item

Index	Name	Description	Is Completed	Action
15	test2	desc2	No	update
16	test3	desc3	No	update
17	test4	desc4	No	update

Delete Item

If we use our injection string ‘0 OR 1=1’, however, all of the contents in the database will be removed.

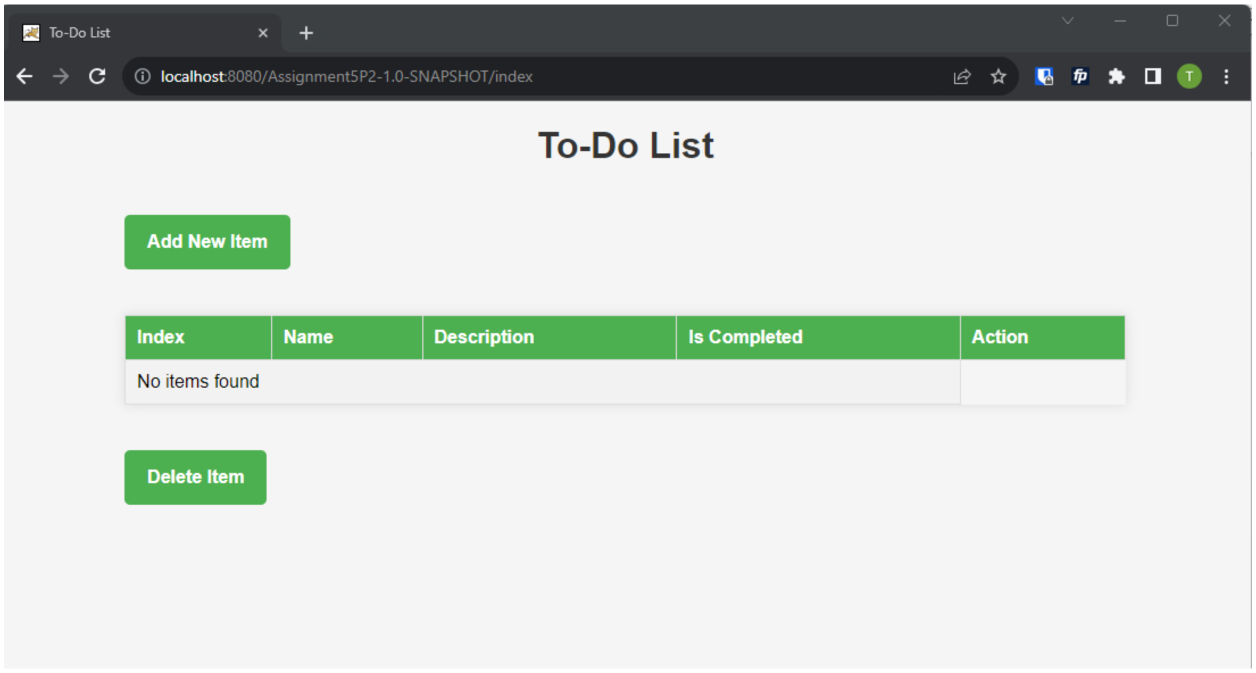


Delete Item

Index to be deleted:

0 OR 1=1

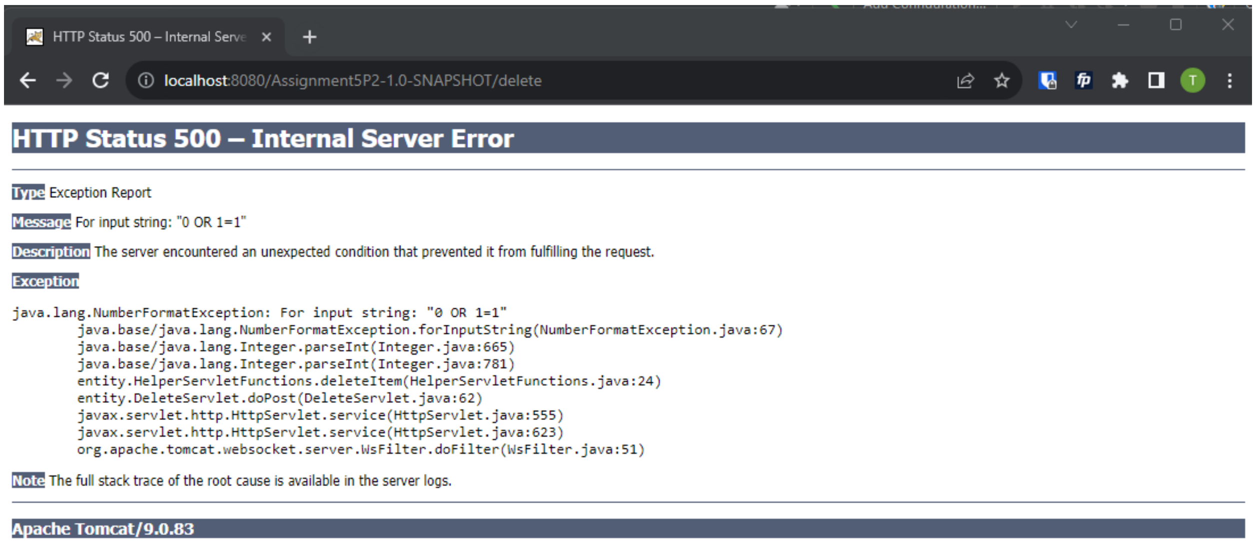
Index	Name	Description	Is Completed
15	test2	desc2	No
16	test3	desc3	No
17	test4	desc4	No



To-Do List

Index	Name	Description	Is Completed	Action
No items found				

As we can see, the injection worked successfully and removed all items from the database. Reverting back to the safe implementation, we get this error when trying to invoke our injection string:


HTTP Status 500 – Internal Serve

localhost:8080/Assignment5P2-1.0-SNAPSHOT/delete

HTTP Status 500 – Internal Server Error

Type Exception Report

Message For input string: "0 OR 1=1"

Description The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception

```
java.lang.NumberFormatException: For input string: "0 OR 1=1"
	java.base/java.lang.NumberFormatException.forInputString(NumberFormatException.java:67)
	java.base/java.lang.Integer.parseInt(Integer.java:665)
	java.base/java.lang.Integer.parseInt(Integer.java:781)
	entity.HelperServletFunctions.deleteItem(HelperServletFunctions.java:24)
	entity.DeleteServlet.doPost(DeleteServlet.java:62)
	javax.servlet.http.HttpServlet.service(HttpServlet.java:555)
	javax.servlet.http.HttpServlet.service(HttpServlet.java:623)
	org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:51)
```

Note The full stack trace of the root cause is available in the server logs.

Apache Tomcat/9.0.83