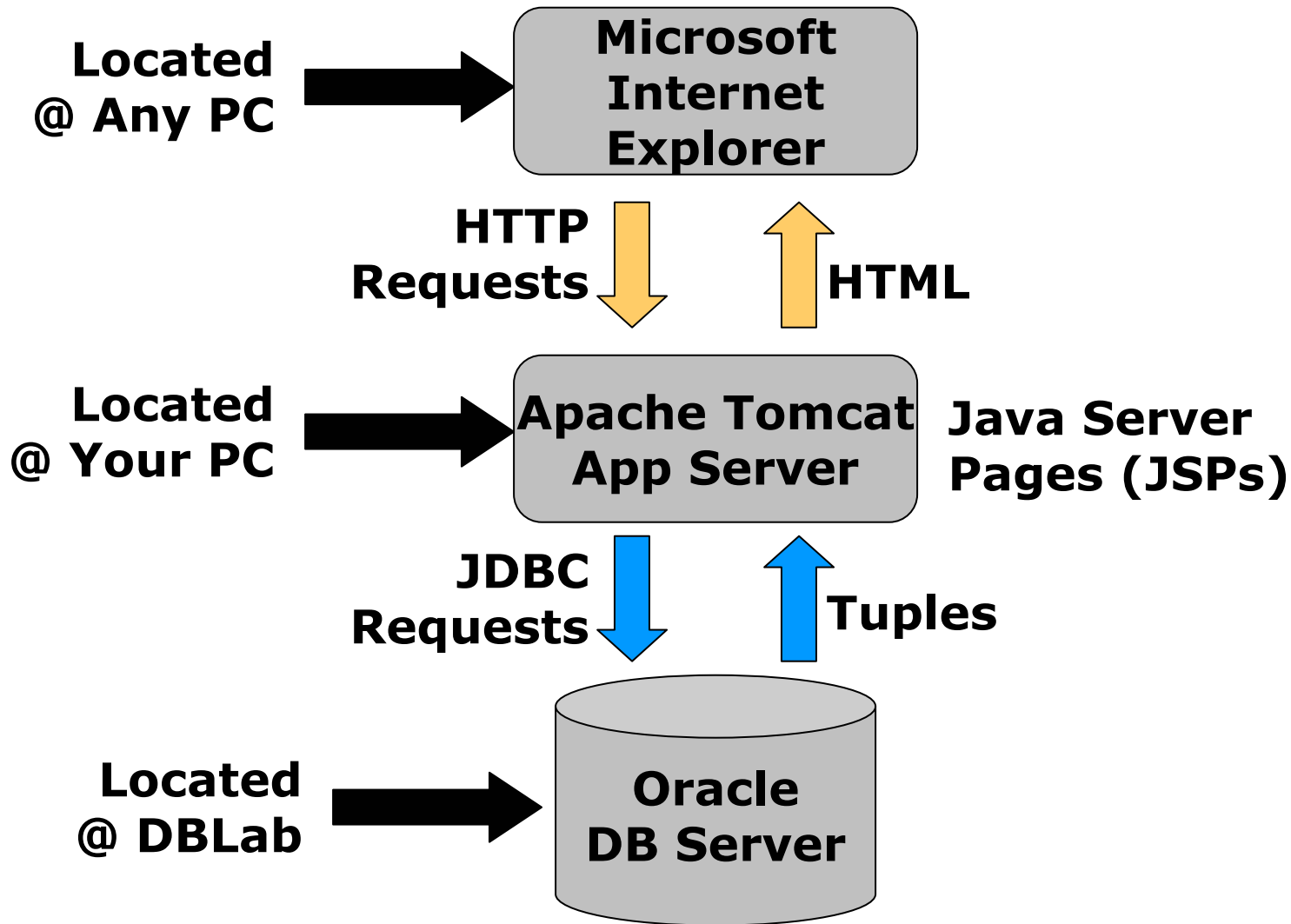


Three-Tier Architecture



Data Entry Forms

The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying `http://michalis:8080/cse132b/students2.jsp`. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The toolbar contains Back, Forward, Stop, Home, Search, Favorites, History, Print, and other icons. The main content area displays a web page titled "Data Entry Menu" with a table of student records and a list of navigation links.

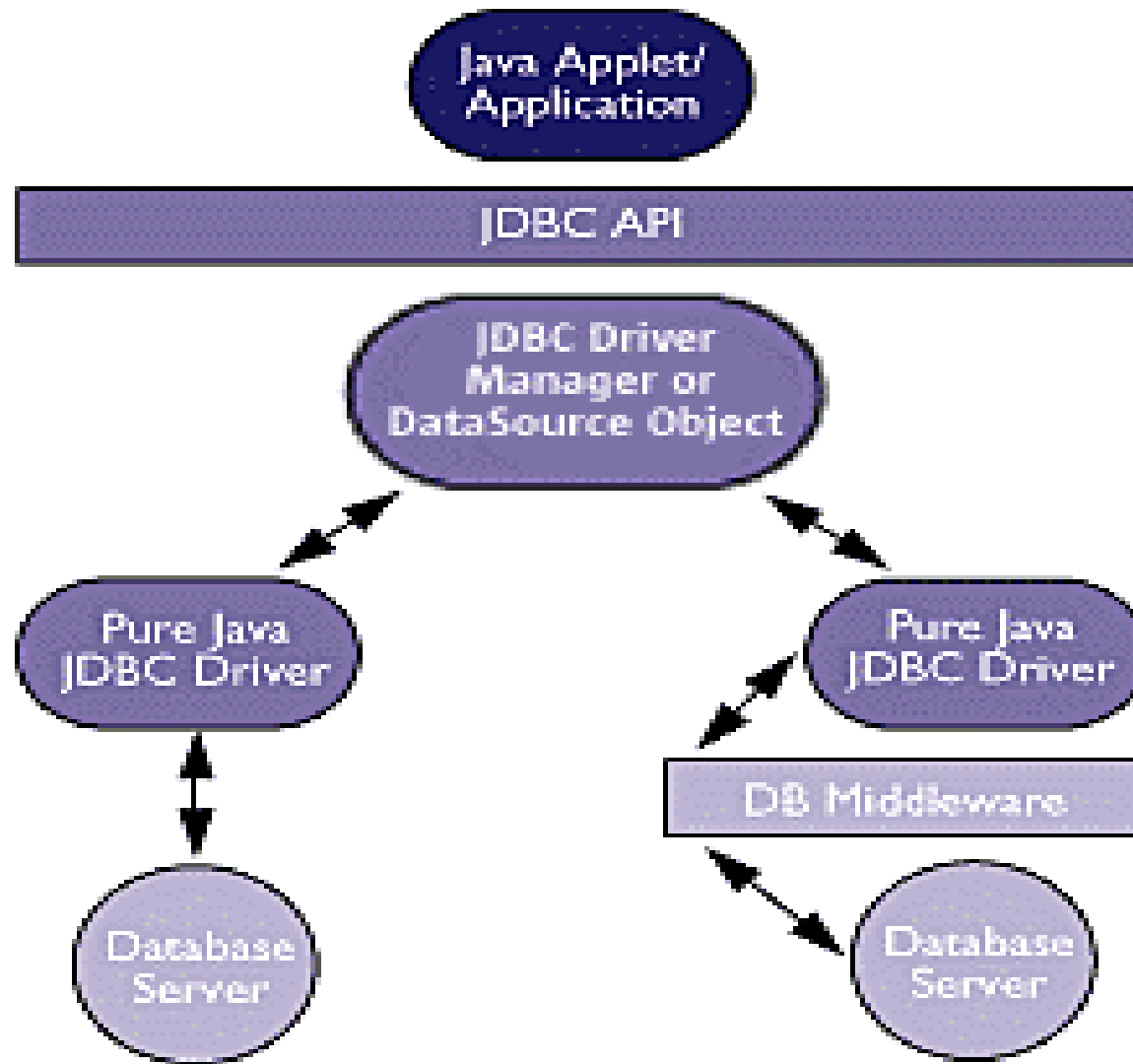
Data Entry Menu

- [Courses](#)
- [Classes](#)
- [Students](#)

SSN	ID	First	Last	College	Action
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="button" value="Insert"/>
123456789	3	Michalis	Petropoulos	Muir	<input type="button" value="Update"/> <input type="button" value="Delete"/>
987654321	2	Vagelis	Hristidis	Muir	<input type="button" value="Update"/> <input type="button" value="Delete"/>

The browser's status bar at the bottom shows "Done" and "Local intranet".

Java Database Connectivity (JDBC)



JDBC

```
import java.sql.*;

class JdbcTest {
public static void main (String args []) throws SQLException {
    // Load SQLServer driver
    DriverManager.registerDriver (new
        com.mssqlserver.jdbc.driver.SQLServerDriver());
    // Connect to the local database
    Connection conn = DriverManager.getConnection
        ("jdbc:sqlserver://kebab.ucsd.edu:1433","scott", "tiger");
```

```
// Query the student names
Statement stmt = conn.createStatement ();
ResultSet rset = stmt.executeQuery ("SELECT name FROM
    Student");
// Print the name out
//name is the 2nd attribute of Student
while (rset.next ())
    System.out.println (rset.getString (2));

//close the result set, statement, and the connection
rset.close();
stmt.close();
conn.close();
```

PreparedStatement Object

If you want to execute a Statement object many times, it will normally reduce execution time to use a PreparedStatement object instead.

```
PreparedStatement updateStud = conn.prepareStatement(
    "UPDATE Student SET name = ? WHERE lastname LIKE
    ?");
```

```
updateStud.setString(1, "John");
updateStud.setString(2, "Smith");
```

```
updateStud.executeUpdate();
```

PreparedStatement Object

the following two code fragments accomplish the same thing:

- Code Fragment 1:

```
String updateString = "UPDATE COFFEES SET SALES = 75  
" + "WHERE COF_NAME LIKE 'Colombian';  
stmt.executeUpdate(updateString);
```

- Code Fragment 2:

```
PreparedStatement updateSales = con.prepareStatement(  
"UPDATE COFFEES SET SALES = ? WHERE COF_NAME  
LIKE ? "); updateSales.setInt(1, 75);  
updateSales.setString(2, "Colombian");  
updateSales.executeUpdate();
```

- **int getInt(int columnIndex)**

Retrieves the value of the designated column in the current row of this ResultSet object as an int in the Java programming language.

- **int getInt(String columnName)**
- **String getString(int columnIndex)**
- **String getString(String columnName)**

Using Transactions

When a connection is created, it is in auto-commit mode. This means that each individual SQL statement is treated as a transaction and will be automatically committed right after it is executed.

```
conn.setAutoCommit(false);
```

```
....
```

```
transaction
```

```
...
```

```
con.commit();
```

```
con.setAutoCommit(true);
```

Using Transactions

example

```
con.setAutoCommit(false);
```

```
PreparedStatement updateSales = con.prepareStatement( "UPDATE  
COFFEES SET SALES = ? WHERE COF_NAME LIKE ?");
```

```
updateSales.setInt(1, 50);
```

```
updateSales.setString(2, "Colombian");
```

```
updateSales.executeUpdate();
```

```
PreparedStatement updateTotal = con.prepareStatement( "UPDATE  
COFFEES SET TOTAL = TOTAL + ? WHERE COF_NAME LIKE ?");
```

```
updateTotal.setInt(1, 50);
```

```
updateTotal.setString(2, "Colombian");
```

```
updateTotal.executeUpdate();
```

```
con.commit();
```

```
con.setAutoCommit(true);
```

Retrieving Exceptions

JDBC lets you see the warnings and exceptions generated by your DBMS and by the Java compiler. To see exceptions, you can have a catch block print them out. For example, the following two catch blocks from the sample code print out a message explaining the exception:

```
try {  
    // Code that could generate an exception goes here.  
    // If an exception is generated, the catch block below  
    // will print out information about it.  
} catch(SQLException ex) {  
    System.err.println("SQLException: " + ex.getMessage());  
}
```

JSP Syntax

- Comment

- `<%-- Comment --%>`

- Expression

- `<%= java expression %>`

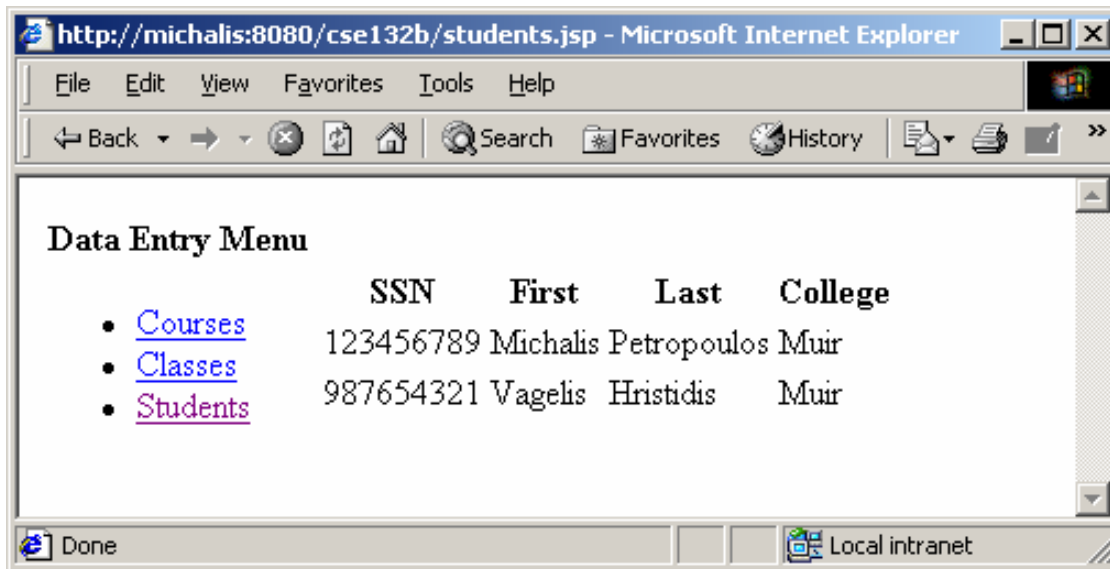
- Scriptlet

- `<% java code fragment %>`

- Include

- `<jsp:include page="relativeURL" />`

Entry Form - First Attempt



Entry Form - First Attempt

Menu HTML Code

```
<b>Data Entry Menu</b>
<ul>
  <li>
    <a href="courses.jsp">Courses<a>
  </li>
  <li>
    <a href="classes.jsp">Classes<a>
  </li>
  <li>
    <a href="students.jsp">Students<a>
  </li>
</ul>
```

Entry Form - First Attempt

JSP Code

```
<html>
<body>
  <table>
    <tr>
      <td>
        <jsp:include page="menu.html" />
      </td>
      <td>
        Open connection code
Statement code
Presentation code
Close connection code
      </td>
    </tr>
  </table>
</body>
</html>
```

Entry Form - First Attempt

Open Connectivity Code

```
<%-- Set the scripting language to java and --%>
<%-- import the java.sql package --%>
<%@ page language="java" import="java.sql.*" %>

<%
    try {
        // Load Oracle Driver class file
        DriverManager.registerDriver
            (new oracle.jdbc.driver.OracleDriver());

        // Make a connection to the Oracle datasource
        Connection conn = DriverManager.getConnection
            ("jdbc:oracle:thin:@feast.ucsd.edu:1521:source",
            "user", "pass");
    }
%>
```


Entry Form - First Attempt

Statement Code

```
<%  
// Create the statement  
Statement statement = conn.createStatement();  
  
// Use the statement to SELECT the student attributes  
// FROM the Student table.  
ResultSet rs = statement.executeQuery  
    ("SELECT * FROM Student");  
%>
```

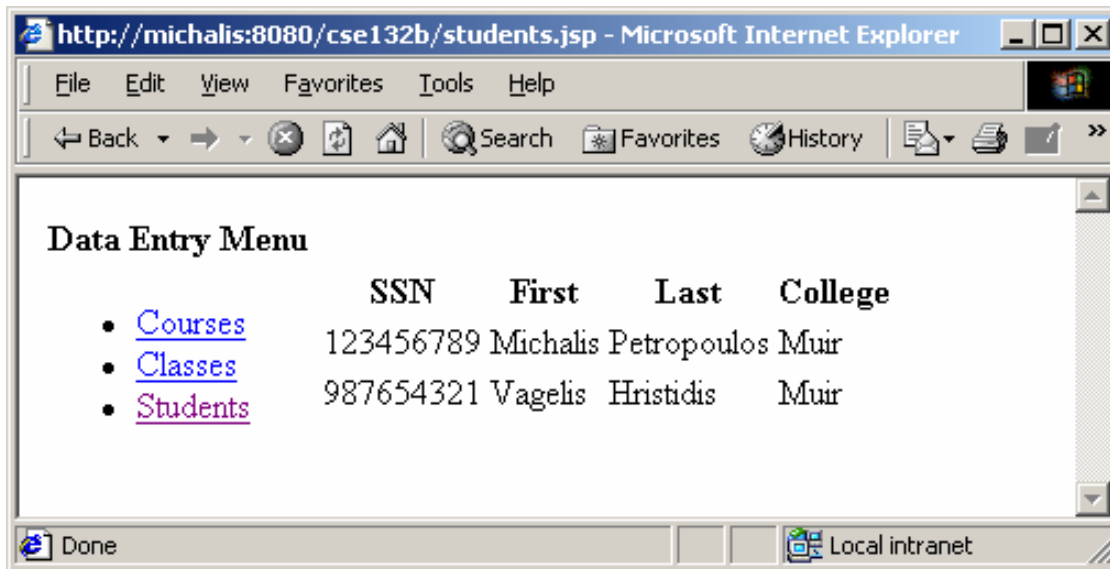
Entry Form - First Attempt

Presentation Code

```
<table>
  <tr>
    <th>SSN</th>
    <th>First</th>
    <th>Last</th>
    <th>College</th>
  </tr>

  <%
    // Iterate over the ResultSet
    while ( rs.next() ) {
  %>
    Iteration Code
  <%
    }
  %>
</table>
```

Entry Form - First Attempt



Entry Form - First Attempt

Iteration Code

```
<tr>
  <%-- Get the SSN, which is a number --%>
  <td><%= rs.getInt("SSN") %></td>

  <%-- Get the ID --%>
  <td><%= rs.getString("ID") %></td>

  <%-- Get the FIRSTNAME --%>
  <td><%= rs.getString("FIRSTNAME") %></td>

  <%-- Get the LASTNAME --%>
  <td><%= rs.getString("LASTNAME") %></td>

  <%-- Get the COLLEGE --%>
  <td><%= rs.getString("COLLEGE") %></td>
</tr>
```

Entry Form - First Attempt

Close Connectivity Code

```
<%  
// Close the ResultSet  
rs.close();  
  
// Close the Statement  
statement.close();  
  
// Close the Connection  
conn.close();  
  
} catch (SQLException sqle) {  
    out.println(sqle.getMessage());  
} catch (Exception e) {  
    out.println(e.getMessage());  
}  
%>
```

Entry Form - Second Attempt

http://michalis:8080/cse132b/students1.jsp - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Back Forward Stop Home Search Favorites History Print Mail News Chat

Data Entry Menu

	SSN	ID	First	Last	College	Action
• Courses	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	Insert
• Classes	123456789	3	Michalis	Petropoulos	Muir	
• Students	987654321	2	Vagelis	Hristidis	Muir	

Done Local intranet

Entry Form - Second Attempt

JSP Code

```
<html>
<body>
  <table>
    <tr>
      <td>
        Open connection code
        Insertion Code
        Statement code
        Presentation code
        Close connection code
      </td>
    </tr>
  </table>
</body>
</html>
```

Entry Form - Second Attempt

Insertion Code

```
// Check if an insertion is requested
String action = request.getParameter("action");
if (action != null && action.equals("insert")) {

conn.setAutoCommit(false);

// Create the prepared statement and use it to
// INSERT the student attrs INTO the Student table.
PreparedStatement pstmt = conn.prepareStatement(
("INSERT INTO Student VALUES (?, ?, ?, ?, ?)"));

pstmt.setInt(1,Integer.parseInt(request.getParameter("SSN")));
pstmt.setString(2, request.getParameter("ID"));
...
pstmt.executeUpdate();

conn.commit();
conn.setAutoCommit(true);
}
```


Entry Form - Second Attempt

Presentation Code

```
<table>
  <tr>
    <th>SSN</th>
    <th>First</th>
    <th>Last</th>
    <th>College</th>
  </tr>
  Insert Form Code
<%
  // Iterate over the ResultSet
  while ( rs.next() ) {
%>
  Iteration Code
<%
  }
%>
</table>
```

Entry Form - Second Attempt

Insert Form Code

```
<tr>
  <form action="students.jsp" method="get">
    <input type="hidden" value="insert" name="action">
    <th><input value="" name="SSN" size="10"></th>
    <th><input value="" name="ID" size="10"></th>
    <th><input value="" name="FIRSTNAME" size="15"></th>
    <th><input value="" name="LASTNAME" size="15"></th>
    <th><input value="" name="COLLEGE" size="15"></th>
    <th><input type="submit" value="Insert"></th>
  </form>
</tr>
```

Entry Form - Third Attempt

The screenshot shows a Microsoft Internet Explorer browser window with the address bar displaying `http://michalis:8080/cse132b/students2.jsp`. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The toolbar contains Back, Forward, Stop, Home, Search, Favorites, History, and other navigation icons. The main content area displays a "Data Entry Menu" with three links: [Courses](#), [Classes](#), and [Students](#). Below the menu is a table with columns for SSN, ID, First, Last, College, and Action. The table contains three rows of data, with the first row being empty and the other two containing student records. The Action column for each row contains "Insert", "Update", and "Delete" buttons respectively.

Data Entry Menu		SSN	ID	First	Last	College	Action
<ul style="list-style-type: none">CoursesClassesStudents							Insert
	123456789	3	Michalis	Petropoulos	Muir		Update Delete
	987654321	2	Vagelis	Hristidis	Muir		Update Delete

The status bar at the bottom of the browser window shows "Done" and "Local intranet".

Entry Form - Third Attempt

JSP Code

```
<html>
<body>
  <table>
    <tr>
      <td>
        Open connection code
        Insertion Code
        Update Code
        Delete Code
        Statement code
        Presentation code
        Close connection code
      </td>
    </tr>
  </table>
</body>
</html>
```

Entry Form - Third Attempt

Update Code

```
// Check if an update is requested
if (action != null && action.equals("update")) {

conn.setAutoCommit(false);

// Create the prepared statement and use it to
// UPDATE the student attributes in the Student table.
PreparedStatement pstatement = conn.prepareStatement(
"UPDATE Student SET ID = ?, FIRSTNAME = ?, " +
"LASTNAME = ?, COLLEGE = ? WHERE SSN = ?");

pstatement.setString(1, request.getParameter("ID"));
pstatement.setString(2, request.getParameter("FIRSTNAME"));
...
int rowCount = pstatement.executeUpdate();

conn.setAutoCommit(false);
conn.setAutoCommit(true);
}
```

Entry Form - Third Attempt

Delete Code

```
// Check if a delete is requested
if (action != null && action.equals("delete")) {

conn.setAutoCommit(false);

// Create the prepared statement and use it to
// DELETE the student FROM the Student table.
PreparedStatement pstmt = conn.prepareStatement(
"DELETE FROM Student WHERE SSN = ?");

pstmt.setInt(1,
    Integer.parseInt(request.getParameter("SSN")));
int rowCount = pstmt.executeUpdate();

conn.setAutoCommit(false);
conn.setAutoCommit(true);
}
```

Entry Form - Third Attempt

Presentation Code

```
<table>
  <tr>
    <th>SSN</th>
    <th>First</th>
    <th>Last</th>
    <th>College</th>
  </tr>
  Insert Form Code
<%
  // Iterate over the ResultSet
  while ( rs.next() ) {
<%
    Iteration Code
<%
  }
<%
</table>
```

Entry Form - Third Attempt

Iteration Code

```
<tr>
  <form action="students.jsp" method="get">
    <input type="hidden" value="update" name="action">
    <td><input value="<%= rs.getInt("SSN") %>" name="SSN"></td>
    <td><input value="<%= rs.getString("ID") %>" name="ID"></td>
  ..
    <td><input type="submit" value="Update"></td>
  </form>
  <form action="students2.jsp" method="get">
    <input type="hidden" value="delete" name="action">
    <input type="hidden" value="<%= rs.getInt("SSN") %>"
    name="SSN">
    <td><input type="submit" value="Delete"></td>
  </form>
</tr>
```