LAZY TABLE

An alternative solution to JSF “dataTable” session overload

Install Guide.

The Lazy Table solution uses the following technologies

- JSF 2.0 (MVC framework for building J2EE based web applications)
- AJAX4JSF
- TOMAHAWK
- JPA
- EJB 3.0

In addition to run the application you would require a database server and an application server.

The steps to install are as follows:

Step 1

Database server: Install one of the free RDBMS servers like PostgresSQL or MySql.

Step 2

Create a database schema and populate the table with data using the script in github under:

https://github.com/plimaye/LazyTag/tree/master/Scripts

If you are using PostgreSQL the script is named:

PostgreSQL_DBScript.txt

If you are using MySQL, the script is named:

MySQL_DBScript.txt

Step 3

Application server and IDE:

We advice you install NetBeans IDE as it comes pre-configured GlassFish Application Server. You are free to install any other IDE and any other application servers as long as you can follow the instructions for configuring the application relevant to the server of your choice.

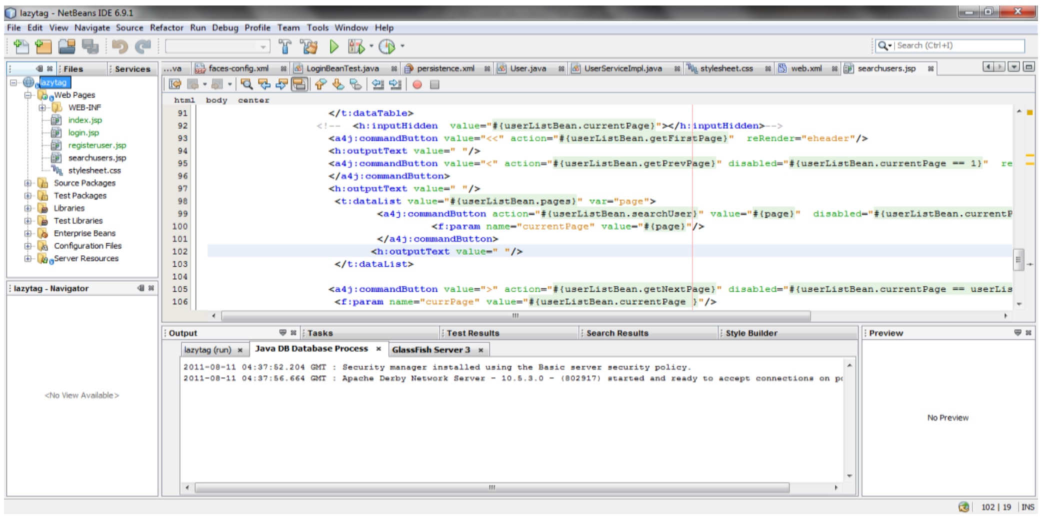
Step 4

Download the source code from the github source link:

https://github.com/plimaye/LazyTag

Step 5

Import the source code folder into the IDE of your choice.


```
</t:dataTable>
<!--  <h:inputHidden  value="#{userListBean.currentPage}"></h:inputHidden>-->
<a4j:commandButton value="<<" action="#{userListBean.getFirstPage}"  reRender="eheader"/>
<h:outputText value=" "/>
<a4j:commandButton value="<" action="#{userListBean.getPrevPage}" disabled="#{userListBean.currentPage == 1}"  re
</a4j:commandButton>
<h:outputText value=" "/>
 <t:dataList value="#{userListBean.pages}" var="page">
        <a4j:commandButton action="#{userListBean.searchUser}" value="#{page}"  disabled="#{userListBean.currentP
                <f:param name="currentPage" value="#{page}"/>
        </a4j:commandButton>
       <h:outputText value=" "/>
 </t:dataList>

<a4j:commandButton value=">" action="#{userListBean.getNextPage}" disabled="#{userListBean.currentPage == userLis
 <f:param name="currPage" value="#{userListBean.currentPage }"/>
```

```
2011-08-11 04:37:52.204 GMT : Security manager installed using the Basic server security policy.
2011-08-11 04:37:56.664 GMT : Apache Derby Network Server - 10.5.3.0 - (802917) started and ready to accept connections on po
```

Step 6

We assume you have the pre-requisites to compile a Java Enterprise Project like JDK6 (JAVA_HOME, environment variable setup).

Step 7

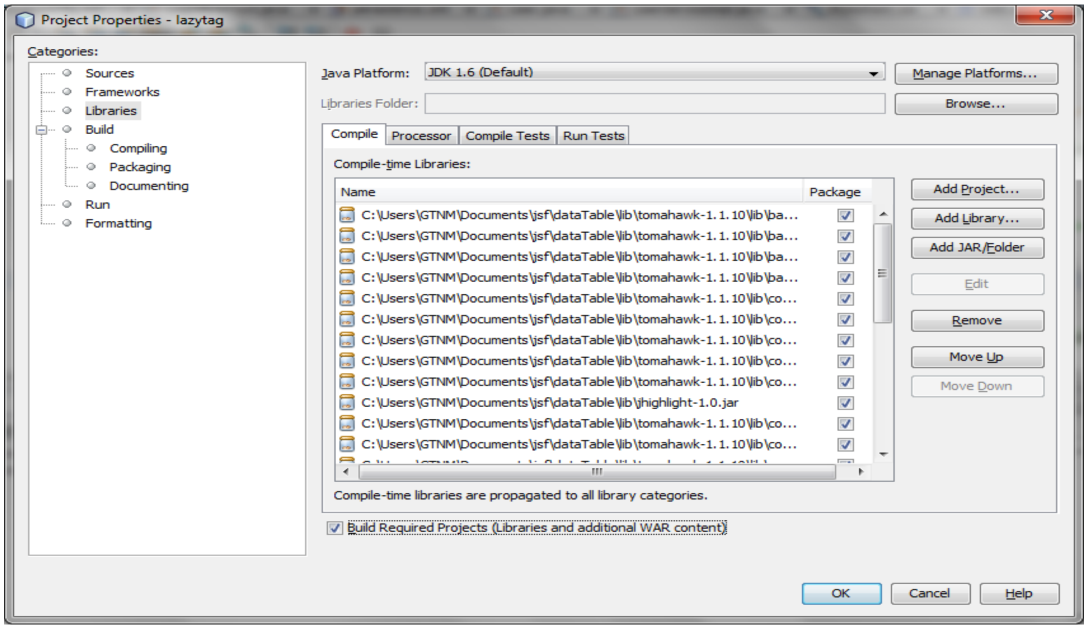
The following jars which are required for the project.

- Database driver jar (specific to the database you choose to connect to) eg: postgresql-9.0-801.jdbc4 (for postgres DB)
- ajax4jsf-1.1.1
- tomahawk-1.1.10-bin

You can find all the jars in this location in our github repository

https://github.com/plimaye/LazyTag/tree/master/Libs

The location of all the above jars needs to be configured in the build path of your project.



Java Platform: JDK 1.6 (Default)

Libraries Folder:

Compile-time Libraries:

Name	Package
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\ba...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\ba...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\ba...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\ba...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\highlight-1.0.jar	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓
C:\Users\GTNM\Documents\jsf\dataTable\lib\tomahawk-1.1.10\lib\co...	✓

Compile-time libraries are propagated to all library categories.

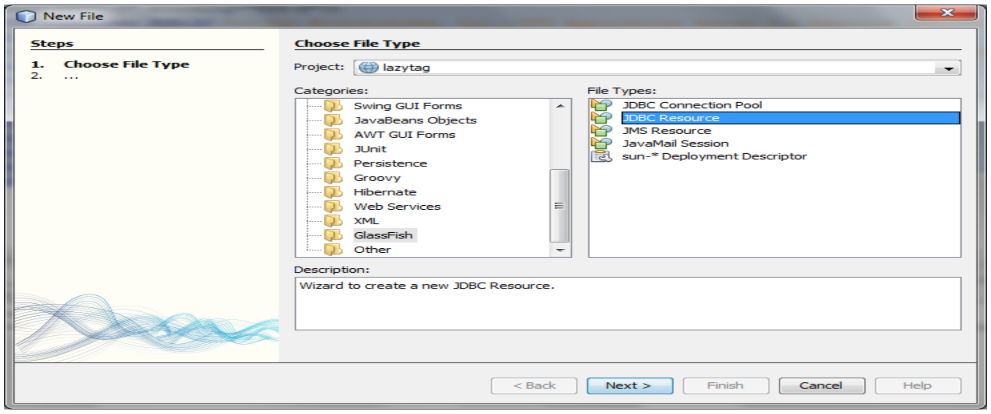
☑ Build Required Projects (Libraries and additional WAR content)

Step 8

Configure the datasource connection details to match the database connection settings. This can be found under the sun-resources.xml file as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE resources PUBLIC "-//Sun Microsystems, Inc.//DTD Application Server
9.0 Resource Definitions //EN"
"http://www.sun.com/software/appserver/dtds/sun-resources_1_3.dtd">
<resources>
  <jdbc-resource enabled="true" jndi-name="jdbc/newDataSource" object-
type="user" pool-name="connectionPool">
    <description/>
  </jdbc-resource>
  <jdbc-connection-pool allow-non-component-callers="false" associate-with-
thread="false" connection-creation-retry-attempts="0" connection-creation-
retry-interval-in-seconds="10" connection-leak-reclaim="false" connection-
leak-timeout-in-seconds="0" connection-validation-method="auto-commit"
datasource-classname="org.postgresql.ds.PGSimpleDataSource" fail-all-
connections="false" idle-timeout-in-seconds="300" is-connection-validation-
required="false" is-isolation-level-guaranteed="true" lazy-connection-
association="false" lazy-connection-enlistment="false" match-
connections="false" max-connection-usage-count="0" max-pool-size="32" max-
wait-time-in-millis="60000" name="connectionPool" non-transactional-
connections="false" pool-resize-quantity="2" res-type="javax.sql.DataSource"
statement-timeout-in-seconds="-1" steady-pool-size="8" validate-atmost-once-
period-in-seconds="0" wrap-jdbc-objects="false">
    <property name="databaseName" value="sampledb" />
    <property name="URL" value="jdbc:postgresql://localhost:5432/sampledb"/>
    <property name="User" value="postgresuser"/>
    <property name="Password" value=" postgrespassword "/>
  </jdbc-connection-pool>
</resources>
```

The above file can be generated by using the : New → File→ Glassfish→JDBC Resource


New File

Steps
1. Choose File Type
2. ...

Choose File Type

Project: lazytag

Categories:
- Swing GUI Forms
- JavaBeans Objects
- AWT GUI Forms
- JUnit
- Persistence
- Groovy
- Hibernate
- Web Services
- XML
- GlassFish
- Other

File Types:
- JDBC Connection Pool
- JDBC Resource
- JMS Resource
- JavaMail Session
- sun-* Deployment Descriptor

Description:
Wizard to create a new JDBC Resource.

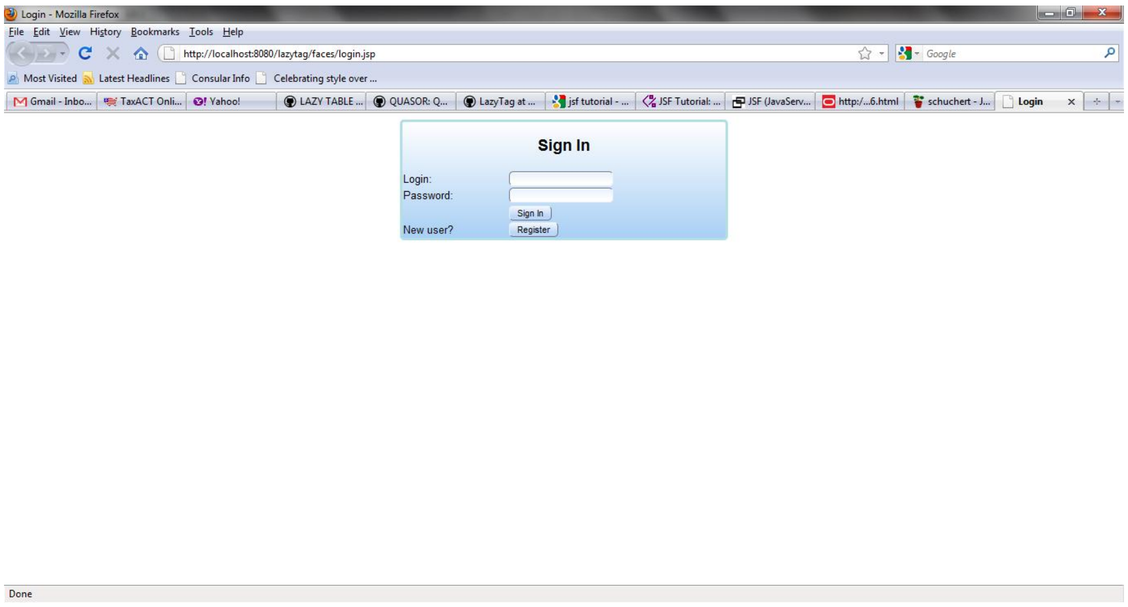
Fill in the required details of the required DB server, driver class, DB connection details.

Step 8

Now that the IDE has been setup, choose to clean and build the project using the IDE.

Step 9

Run the project , which will invoke the login url.


Login - Mozilla Firefox

http://localhost:8080/lazytag/faces/login.jsp

Sign In

Login:
Password:

New user?

Done