Implementing the Model-View-Controller Design Pattern Using Struts

The MVC Design Pattern and its Struts Implementation

What is the MVC Design Pattern?
The MVC design pattern is the formalization of a Web programming approach where the Web application is divided into three component areas:

- The **Model** captures the application’s data and interaction with the data (business logic). The model part owns the database.
- The **View** is the part of the application that produces the HTML pages returned to the user.
- The **Controller** manages the user’s requests.

The typical MVC pattern is as follows: The browser’s http request is received by a controller servlet that (perhaps in cooperation with some controller beans) interacts with beans that comprise the model and interact with the database. Then the controller invokes the jsp’s that comprise the view. Notice that the model beans typically produce data that are displayed by the jsp’s that implement the view.

How can I easily tell which part of my code is which?
Whenever you build a Web application consider a corresponding desktop application. If some data and code are common to both the desktop and the web application, then they should be in the model. If a piece of code outputs HTML it is part of the view. If a piece of code controls the flow across pages then it is likely piece of the controller.

How does Struts facilitate the implementation of MVC?
Struts provides two key areas of support: First it provides a “dispatcher” that allows one to easily build the controller. Second it provides support for form beans, which facilitate the management of html forms.

How to build a Controller using Struts?
The following discussion will use a Graduate Admissions database as the running example. We assume that you have placed the application under the GradAdmissions directory under webapps. The relevant files of source code are provided at the end of the document.

Directing Request Traffic at the Struts **ActionServlet**
The Struts library provides the **ActionServlet** that assumes the role of the controller servlet in a Struts implementation.

To direct the http request traffic to the **ActionServlet** include the following in the web.xml:
1. In the action servlet configuration part of web.xml include the following lines that declare that the servlet named action is the org.apache.struts.action.ActionServlet. You should place the struts.jar, which includes the ActionServlet, in the WEB-INF/lib directory.

```xml
<servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
        <param-name>config</param-name>
        <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <init-param>
        <param-name>debug</param-name>
        <param-value>2</param-value>
    </init-param>
    <init-param>
        <param-name>detail</param-name>
        <param-value>2</param-value>
    </init-param>
    <load-on-startup>2</load-on-startup>
</servlet>
```

2. Then define the following servlet mapping that directs all requests of the form *.do to the action servlet.

```xml
<servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
</servlet-mapping>
```

However, you may not want the users to have to invoke a URL that ends with “.do”. Let us assume that you want the users to invoke the application by simply typing www.myhost.com/GradAdmissions. Then the application server will invoke the index.html or index.jsp you have in the GradAdmissions directory. You can direct the request to the action servlet by using the redirect tag of the jstl core library. Here is a complete index.jsp that does the trick.

```jsp
<%@ taglib uri="jstl/c" prefix="c" %>
<c:redirect url="/index.do"/>
```

The index.do will be passed by the application server to the action servlet.

**Directing the Request Traffic to Controller Beans**

All the *.do URL’s are received by the action servlet, which consults the configuration file struts-config.xml in order to route the request to an appropriate controller bean. In this way you can split the workflow management code among multiple controller beans. The struts-config.xml file is found at the WEB-INF directory. Let us see the entry that directs the index.do request to the controller bean
edu.ucsd.cse.GradAdmissions.Actions.IndexAction, which is under WEB-INF/classes. Within the action-mappings definitions you will find the entry:

```xml
<action
    path="/index"
    type="edu.ucsd.cse.GradAdmissions.Actions.IndexAction">
    <forward name="adminHome" path="/admin/index.jsp" />
    <forward name="chairHome" path="/chair/index.jsp" />
    <forward name="facultyHome" path="/faculty/index.jsp" />
    <forward name="allHome" path="/all/index.jsp" />
</action>
```

The path attribute specifies the entries that will be directed to the bean provided by the type attribute. The forward elements provide aliases (given by the name attribute) for the view jsp’s that may be chosen for displaying the HTML pages.

**A Typical Controller Bean**

First let us discuss some things that have happened already. By the time index.do is called we have already gone thru the log-in page, where the user has given his name and password. We will not discuss yet the implementation of the secure log-in mechanism. However, let us note that by the time that IndexAction is called there is already a State bean, which is part of the model. We have made this bean available via a stateBean property of the session object. The State bean is meant to contain the data of the user who just logged in. Its username property is already set by the log-in process.

The IndexAction bean is responsible for forwarding to the appropriate jsp, upon recognizing the role of the user who just logged in. In particular, if the user is a faculty member the bean will forward to the facultyHome jsp, if it is the committee chair it will forward to chairHome, and so on.

The IndexAction bean, as all controller beans, extends the Action class of Struts. For every controller bean we have to implement a public execute() whose parameters are

- mapping: this is the Java object encoding of the action-mapping data we provided in the struts-config.xml entry that directed us to IndexAction.
- form: we’ll discuss this extensively later.
- request: is the well-known request object provided by the app server.
- response: is the response object provided by the app server, which is rarely used unless one produces HTML by writing directly a servlet.

The first job of the controller bean IndexAction is to find the session object and retrieve its stateBean attribute, which provides the State bean. Notice that getAttribute() function returns an untyped Object. Hence we need to cast the state object in its correct type, which is State.
Next the IndexAction bean forwards to the appropriate jsp. In order to do so, it needs to know what is the role of the user. The information necessary for finding out the role is in the database. In accordance with the MVC design pattern we encapsulate the database lookup functionality in model beans – in this case the State bean provides a function getUserRole() that returns the role of the user and the controller bean forwards to the appropriate jsp.

A few things about how the jsp brings us back to the ActionServlet

We assume in the following discussion that the role of the user is “Admin” and hence the "/admin/index.jsp" is displayed. This jsp includes a header_body.jsp, which in turns includes a menu.jsp, which presents the menu of hyperlinks we see on the left side of the page. (For brevity we do not include those jsp’s in the listing.) The hyperlinks are of the form *.do. Hence, choosing one of them will bring us back to the ActionServlet dispatcher. Let us assume that we click on “Status”. The associate hyperlink is /all/summary.do. In the struts-config.xml there is the following action mapping, which will forward the request to the edu.ucsd.cse.GradAdmissions.Actions.SummaryAction controller bean.

```
<action path="/all/summary"
        type="edu.ucsd.cse.GradAdmissions.Actions.SummaryAction">
     <forward name="summary" path="/all/overallSummary.jsp" />
</action>
```

And a Second Triplet of Controller Bean, Model Bean and View JSP

The SummaryAction controller bean cooperates with the Summary model bean, which is responsible for providing the data for a summary of the admission process status, as shown in this page.

The SummaryAction bean is straightforward: It obtains the Summary object from the session, if there is already a Summary object. Otherwise it creates a new Summary object. We do not list the Summary object since its details are irrelevant to the MVS pattern. Consequently SummaryAction invokes the getSummary() function of the Summary object. This function returns an object of type (sql query) Result, which has the data shown on the html page. Recall, the Result object is essentially a series of rows of untyped objects, which can be addressed by column name or column number. The key aspect of the MVC model is that while we code the controller we should not worry about the SQL that produces the Result. It suffices to know that the Summary object returns the right data when we invoke its getSummary() function.

Forms in Struts

Struts allows one to build form beans that are in close interaction with the forms and allow the easy transfer of data between the Java code of the controller beans and the jsp pages.
Let us consider the following example: Upon clicking the “Faculty” link of the menu the URL “/admin/faculty.do” is invoked and we see a page, which allows us to edit and delete faculty members (see attached figure at the end of the document – the right side of the page where the “Remove” buttons are is cut). Upon clicking the “Edit” of, say, the first row we see the “faculty edit” page you see at the end of the document.

**Form Bean Declaration in struts-config.xml**

Let’s see how we organized controller beans, jsp’s and form beans to achieve those pages. The struts-config.xml file contains the following action mapping.

```xml
<action path="/admin/faculty"
   type="edu.ucsd.cse.GradAdmissions.Actions.FacultyAction"
   name="facultyForm"
   input="/admin/facultyEdit.jsp"
   scope="request">
   <forward name="faculty" path="/admin/faculty.jsp" />
</action>
```

The new features to notice are

- The form bean FacultyForm is associated with the controller bean FacultyAction. Those two beans will be in charge of coordinating the forms and the flow we discussed above, respectively. The scope of the facultyForm bean is “request”, which means that a new facultyForm object will be created for each request and the old will be destroyed.
- The jsp facultyEdit.jsp will be responsible for displaying the form associated with the facultyForm bean.

Notice that the controller bean and the form bean are responsible for more than one kind of pages. This is a practice that though it should not be overdone (after all, your whole web application may have only one controller bean – this would be bad) it is OK to do when the pages share a lot of the same function and interact with the same model bean.

**First run of the controller bean**

The ActionServlet dispatcher upon receiving the “/admin/faculty.do” URL it invokes the FacultyAction bean and a FacultyForm object is passed to it in its form parameter.

Since the FacultyForm bean accommodates all of the “Edit”, “Remove”, “Add New”, “Update” (this is in the facultyEdit.jsp) actions the object provides a `getAction()` function that tells the controller bean whether the action was “Edit”, “Delete”, etc. In the first invocation of the controller bean facultyAction the action is null since no form has been completed yet.
The model bean that provides database functionality to the controller is the `Faculty` bean. The controller bean invokes the `getAllFaculty()` function to obtain a `Result` that lists all faculty members. Consequently it invokes the jsp aliased as `faculty`, which is the “/admin/faculty.jsp” according to the action mapping above. The jsp displays the list of faculty. The interesting part to note in the jsp is how the HTML forms are connected to the form bean, as explained next.

**Including forms in the jsp and connecting them to form beans**

Look for the `html:form` tags in the `faculty.jsp`. The following one is responsible for the “Add New” button. For simplicity we have excluded some html material that appeared on the jsp.

```html
<html:form action="/admin/faculty">
  <html:hidden property="action" value="Add" />
  <html:submit styleClass="body2">
    <bean:message key="button.Add"/> //this simply prints “Add New”
  </html:submit>
</html:form>
```

The `action` attribute specifies the action mapping (found in `struts-config.xml`) that provides the controller bean and the form bean that will be used if this form is used. Notice that since the `facultyForm` bean has “request” scope a new object will be created upon submitting the form. Hence you should not count on any property of the form bean to be already set by prior invocations of the bean. You should instead set all relevant values. The `html:hidden` tag sets the `action` property of the form bean to “Add”. Obviously this requires that the particular form bean provides a function `setAction()`.

Similarly, for each row of the faculty table we have two `html:form` elements that provide the “Edit” and “Remove” buttons. The difference to notice is that the `facultyId` property of the form bean is also set, so we know which faculty member we chose to edit or delete.

Let us assume that the user clicks the “Edit” for the Scott Baden. This will bring us back to the `facultyAction` controller, whereas a `facultyForm` bean will have been created, where the `action` property will be “Edit” and the `facultyId` will be the one of Scott Baden.

We want to present to the user the form of "/admin/facultyEdit.jsp", where we already have completed Scott’s data and we prompt for editing. Let us see how we achieve populating the fields of the form (before the user edits it) by passing information from the controller bean to the form via the form bean:

- Notice the code under “if (action.equals("Edit"))” in the controller bean. Using the `getSingleFaculty()` function of the state bean `faculty` we obtain a result that has a single row containing the data of the faculty identified by
facultyForm.getFacultyID(), i.e., of the faculty that was just chosen by the user.

- We use the data of the row to set corresponding properties (such as first name, last name, etc) of the form bean.
- Finally, the getInputForward() call will forward to the "/admin/facultyEdit.jsp", which is responsible for displaying the data of the form bean.

html: text within the context of html: form

The facultyEdit.jsp contains input text boxes and input check boxes that are preset with the corresponding properties of the form bean, after they are edited they write back their content to the properties of the form bean. Here is a relevant piece of code from facultyEdit.jsp:

```html
<html:form action="/admin/faculty">
<tr>
<td class="formheader">
<b>First Name</b>:
</td>
<td class="formfield">
<html:text styleClass="body2" property="firstName" maxlength="80" size="35"/>
</td>
</tr>
</html:form>
```

Notice that the property attribute of the html:text element specifies that what is displayed in this text box is the content of the firstName property of the form bean facultyForm and, vice versa, whatever we write in this form will be written back to facultyForm. But how does the html:text know which form bean has the property firstName? The answer is that the html:text is within an html:form that specifies the action admin/faculty and the action mappings in struts-config tell us that the form bean associated with admin/faculty is the facultyForm. The moral of the story: you html:text has to be within html:form.

And some JSTL parenthesis

Another feature to notice in the html:form elements of the facultyEdit.jsp is the use of the faculty model bean to print the name of the second column. It is really unnecessary use, since we know that the column will be “Last Name” but it’s there to remind you of jstl. Notice the jstl element

```html
<c:out value="${facultyBean.faculty.columnNames[2]}"/>
```

A lot of stuff is done by this command, under the hood. First, we look up for the facultyBean parameter in request, and if it is not found in request we look it up in session and if it is not found there either we look it up in application. In our web application the faculty object will be found in the session. Consequently the
".faculty" means that we invoke the getFaculty() property of it, which will return a Result. Finally we produce the list of columns of the result and pick the second one. I could have just as well written “Last Name” but I thought you may want to know the mysteries of jstl!

**Going Back to the Controller Bean**

Upon successfully completing the forms of facultyEdit.jsp we return to FacultyAction (since the action attribute of the html:form element indicated so) and we run the following code.

```java
if (action.equals("Update")) {
    result =
        faculty.updateFaculty(
            DBConnections.getDBConnection(request),
            facultyForm.getFirstName(),
            facultyForm.getLastName(),
            facultyForm.getEmail(),
            facultyForm.getActive(),
            facultyForm.getFacultyID());

    faculty.getAllFaculty(DBConnections.getDBConnection(request));
    ActionMessages messages = new ActionMessages();
    ActionMessage message = new ActionMessage("message.update.success");
    messages.add(ActionMessages.GLOBAL_MESSAGE, message);
    saveMessages(request, messages);
    return mapping.findForward("faculty");
}
```

**Form Validation**

One of the features offered by Struts is the ability to systematically validate forms. For example, you want an email in facultyEdit.jsp to be of the form string@string and you want the submitted first and last name to be non-null strings.

**The validate() function**

In Struts you can provide a public validate() function in your form bean(s). This function will be invoked upon the submission of the form. Let us see the validate() function of the FacultyForm bean. The inputs are the action mapping we provided in the struts-config.xml and the request and the output of the function is a (hopefully empty) list of errors. By the time the validation function is called the properties of the bean have already been set by the form. In our example we check that the submitted first and last name strings are non-empty and the provided email matches the pattern "^\S+@\S+$" (check the documentation of java.util.regex.Pattern to see why this essentially checks that the submitted email is of the form “string@string”). In each case corresponding error messages are added to the errors output object.
Displaying Error Messages and Not Only

If the validation produces errors then the jsp associated with the form bean will be invoked. In our example, if an error is made upon completing the form associated with FacultyForm the facultyEdit.jsp will be invoked again. Notice that we have included in the facultyEdit.jsp the following logic:messages element that presents the error messages.

```xml
<logic:messagesPresent>
  <html:messages id="error">
    <p class="errormessage">
      <c:out value="\${error}" escapeXml="false" />
    </p>
  </html:messages>
</logic:messagesPresent>
```

On a parenthetical remark, notice the attribute escapeXml whose false setting directs the c:out to output any XML/HTML tags as if they are tags – rather than try to output them verbatim.

While we are at reporting error messages, you may see that we can just as well produce non-error messages, like the “successful update” message you see following an update. This message was produced by the following piece of code in faculty.jsp.

```xml
<logic:messagesPresent message="true">
  <html:messages id="message" message="true">
    <p class="infomessage">
      <c:out value="\${message}" escapeXml="false" />
    </p>
  </html:messages>
</logic:messagesPresent>
```

The message was created by the following code in facultyAction.

```java
ActionMessages messages = new ActionMessages();
ActionMessage message = new ActionMessage("message.update.success");
messages.add(ActionMessages.GLOBAL_MESSAGE, message);
saveMessages(request, messages);
```
<web-app>

<!-- General Web Application Description -->
<display-name>CSE Department Graduate Admissions</display-name>

<!-- Context Initialization Parameters -->
<context-param>
    <param-name>webmaster</param-name>
    <param-value>mpetropo@cs.ucsd.edu</param-value>
</context-param>
<context-param>
    <param-name>Faculty</param-name>
    <param-value>faculty</param-value>
</context-param>
<context-param>
    <param-name>Staff</param-name>
    <param-value>staff</param-value>
</context-param>
<context-param>
    <param-name>Chair</param-name>
    <param-value>chair</param-value>
</context-param>
<context-param>
    <param-name>Committee</param-name>
    <param-value>committee</param-value>
</context-param>
<context-param>
    <param-name>Admin</param-name>
    <param-value>admin</param-value>
</context-param>

<!-- Filters -->
<filter>
    <filter-name>State Filter</filter-name>
    <filter-class>edu.ucsd.cse.GradAdmissions.Filters.StateFilter</filter-class>
</filter>

<!-- Filter Mappings -->
<filter-mapping>
    <filter-name>State Filter</filter-name>
    <url-pattern>*.do</url-pattern>
</filter-mapping>

<!-- Action Servlet Configuration -->
<servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-class>
    <init-param>
        <param-name>config</param-name>
    </init-param>
</servlet>
<param-value>/WEB-INF/struts-config.xml</param-value>
</init-param>
<init-param>
  <param-name>debug</param-name>
  <param-value>2</param-value>
</init-param>
<init-param>
  <param-name>detail</param-name>
  <param-value>2</param-value>
</init-param>
<load-on-startup>2</load-on-startup>
</servlet>

<!-- Action Servlet Mapping -->
<servlet-mapping>
  <servlet-name>action</servlet-name>
  <url-pattern>*.do</url-pattern>
</servlet-mapping>

<!-- Default Session Timeout in Minutes -->
<session-config>
  <session-timeout>0</session-timeout>
</session-config>

<!-- Mappings Between Extensions and a Mime Types -->
<mime-mapping>
  <extension>pdf</extension>
  <mime-type>application/pdf</mime-type>
</mime-mapping>

<!-- The Welcome File List -->
<welcome-file-list>
  <welcome-file>index.jsp</welcome-file>
</welcome-file-list>

<!-- The Error Pages -->

<!-- Struts Tag Library Descriptors -->
<taglib>
  <taglib-uri>struts/bean-el</taglib-uri>
  <taglib-location>/WEB-INF/lib/taglib/struts-bean-el.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>struts/html-el</taglib-uri>
  <taglib-location>/WEB-INF/lib/taglib/struts-html-el.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>struts/logic-el</taglib-uri>
  <taglib-location>/WEB-INF/lib/taglib/struts-logic-el.tld</taglib-location>
</taglib>
<taglib>
  <taglib-uri>jstl/c</taglib-uri>
  <taglib-location>/WEB-INF/lib/taglib/c.tld</taglib-location>
</taglib>
<resource-ref>
  <description>
  Resource reference to the factory for java.sql.Connection 
  instances that is used for talking to GradAdmissionsDB 
  database that is configured in the server.xml file.
  </description>
  <res-ref-name>jdbc/GradAdmissionsDBAdmin</res-ref-name>
  <res-type>javax.sql.DataSource</res-type>
  <res-auth>Container</res-auth>
</resource-ref>

<!-- Security Constraints -->
<security-constraint>
  <display-name>Admin Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>Admin Area</web-resource-name>
    <url-pattern>/admin/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>admin</role-name>
  </auth-constraint>
  <user-data-constraint>
    <description>SSL Required</description>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<security-constraint>
  <display-name>Committee Chair Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>Committee Chair Area</web-resource-name>
    <url-pattern>/chair/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>chair</role-name>
  </auth-constraint>
  <user-data-constraint>
    <description>SSL Required</description>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<security-constraint>
  <display-name>Faculty Members Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>Faculty Members Area</web-resource-name>
    <url-pattern>/faculty/*</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>admin</role-name>
    <role-name>faculty</role-name>
    <role-name>chair</role-name>
  </auth-constraint>
</security-constraint>
<role-name>committee</role-name>
</auth-constraint>
<user-data-constraint>
  <description>SSL Required</description>
  <transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
</security-constraint>

<security-constraint>
  <display-name>Faculty and Staff Members Security Constraint</display-name>
  <web-resource-collection>
    <web-resource-name>Faculty and Staff Members Area</web-resource-name>
    <!-- Define the context-relative URL(s) to be protected -->
    <url-pattern>/all/*</url-pattern>
    <url-pattern>*.jsp</url-pattern>
    <url-pattern>*.do</url-pattern>
  </web-resource-collection>
  <auth-constraint>
    <role-name>faculty</role-name>
    <role-name>staff</role-name>
    <role-name>chair</role-name>
    <role-name>committee</role-name>
    <role-name>admin</role-name>
  </auth-constraint>
  <user-data-constraint>
    <description>SSL Required</description>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>

<!-- Login configuration uses form-based authentication -->
<login-config>
  <auth-method>FORM</auth-method>
  <realm-name>GradAdmissions Form-Based Authentication Area</realm-name>
  <form-login-config>
    <form-login-page>/login.jsp</form-login-page>
    <form-error-page>/loginerror.jsp</form-error-page>
  </form-login-config>
</login-config>

<!-- Security roles referenced by this web application -->
<security-role>
  <role-name>faculty</role-name>
</security-role>
<security-role>
  <role-name>staff</role-name>
</security-role>
<security-role>
  <role-name>chair</role-name>
</security-role>
<security-role>
  <role-name>committee</role-name>
</security-role>
<role-name>admin</role-name>
</security-role>

</web-app>
struts-config.xml

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE struts-config PUBLIC
 "-//Apache Software Foundation//DTD Struts Configuration
 1.1//EN"
 "http://jakarta.apache.org/struts/dtds/struts-config_1_1.dtd">

<struts-config>

<!-- ========= Data Source Configuration ======== -->
<data-sources />

<!-- ========= Form Bean Definitions ========= -->
<form-beans>
  <form-bean
      name="admissionsYearsForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.AdmissionsYearsForm" />
  <form-bean
      name="facultyForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.FacultyForm" />
  <form-bean
      name="staffForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.StaffForm" />
  <form-bean
      name="editCommitteeForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.EditCommitteeForm" />
  <form-bean
      name="editCommitteeAssignmentsForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.EditCommitteeAssignmentsForm" />
  <form-bean
      name="editFacultyAssignmentsForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.EditFacultyAssignmentsForm" />
  <form-bean
      name="applicantsForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.ApplicantsForm" />
  <form-bean
      name="applicantsReviewsForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.ApplicantsReviewsForm" />
  <form-bean
      name="editCommitteeForm"
      type="edu.ucsd.cse.GradAdmissions.ActionForms.EditCommitteeForm" />
  <form-bean
      name="queueSummaryForm"
<global-exceptions>
<exception
    key="exception.application"
    type="edu.ucsd.cse.GradAdmissions.ApplicationException"
    path="/error.jsp" />
<exception
    key="exception.database"
    type="edu.ucsd.cse.GradAdmissions.DatabaseException"
    path="/error.jsp" />
<exception
    key="exception.security.role"
    type="edu.ucsd.cse.GradAdmissions.SecurityException"
    path="/error.jsp" />
</global-exceptions>

<global-forwards>
    <forward name="error" path="/error.jsp" />
    <forward name="queue" path="/faculty/queue.do?action=remaining" />
</global-forwards>

<action-mappings>
    <action
        path="/index"
        type="edu.ucsd.cse.GradAdmissions.Actions.IndexAction">
        <forward name="adminHome" path="/admin/index.jsp" />
        <forward name="chairHome" path="/chair/index.jsp" />
        <forward name="facultyHome" path="/faculty/index.jsp" />
        <forward name="allHome" path="/all/index.jsp" />
    </action>
    <action
        path="/logout"
        type="edu.ucsd.cse.GradAdmissions.Actions.LogoutAction">
        <forward name="logout" path="/index.do" redirect="true" />
    </action>
    <action
        path="/admin/admissionsYears"
        type="edu.ucsd.cse.GradAdmissions.Actions.AdmissionsYearsAction"
        name="admissionsYearsForm"
        input="/admin/admissionsYears.jsp"
        scope="request">
    </action>
    <action
        path="/admin/faculty"
        type="edu.ucsd.cse.GradAdmissions.Actions.FacultyAction"
        name="facultyForm"
        input="/admin/facultyEdit.jsp"
        scope="request">
    </action>
</action-mappings>
<!-- ========= Message Resources Definitions ===== -->
<message-resources
parameter="edu.ucsd.cse.GradAdmissions.ApplicationResources"/>

<!-- ========= Plug Ins Configuration ===== -->
</struts-config>
IndexAction.java
package edu.ucsd.cse.GradAdmissions.Actions;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import edu.ucsd.cse.GradAdmissions.Constants;
import edu.ucsd.cse.GradAdmissions.SecurityException;

/**
 * Implementation of <strong>Action</strong> that determines which
 * home page to forward to depending on the role of the currently
 * logged on user. Can be <code>Admin</code>, <code>Chair</code>,
 * <code>Committee</code>, <code>Faculty</code> or <code>Staff</code>.
 * *
 */
public class IndexAction extends Action {

/**
 * Process the specified HTTP request, and determines which
 * home page to forward to depending on the role of the currently
 * logged on user. Can be <code>Admin</code>, <code>Chair</code>,
 * <code>Committee</code>, <code>Faculty</code> or <code>Staff</code>.
 * *
 * @param mapping The ActionMapping used to select this instance
 * @param actionForm The optional ActionForm bean for this request (if any)
 * @param request The HTTP request we are processing
 * @param response The HTTP response we are creating
 * @exception SecurityException if user's role is not recognized
 * @return ActionForward representing the home page of the corresponding
 * user role
 */
public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {

        HttpSession session;
        State state = null;
        boolean result;

        session = request.getSession();
        // get the State bean, notice that stateObj is untyped
        Object stateObj = session.getAttribute("stateBean");
        if (stateObj == null || !(stateObj instanceof State)) {
            throw new SecurityException(request.getRemoteUser());
        } else {
            // cast stateObj in its correct type (State)
            state = (State) stateObj;
        }

        // ...
if (state.getUserRole().equals("Faculty"))
    return mapping.findForward("facultyHome");
if (state.getUserRole().equals("Staff"))
    return mapping.findForward("allHome");
if (state.getUserRole().equals("Chair"))
    return mapping.findForward("chairHome");
if (state.getUserRole().equals("Committee"))
    return mapping.findForward("facultyHome");
if (state.getUserRole().equals("Admin"))
    return mapping.findForward("adminHome");

    throw new SecurityException(request.getRemoteUser());
SummaryAction.java
package edu.ucsd.cse.GradAdmissions.Actions;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import edu.ucsd.cse.GradAdmissions.Constants;
import edu.ucsd.cse.GradAdmissions.DBConnections;

public class SummaryAction extends Action {
    public ActionForward execute(
        ActionMapping mapping,
        ActionForm form,
        HttpServletRequest request,
        HttpServletResponse response)
        throws Exception {
            HttpSession session;
            Summary summary = null;
            boolean result;

            session = request.getSession();
            summary = (Summary) session.getAttribute(Constants.SUMMARY_KEY);
            if (summary == null || !(summary instanceof Summary)) {
                summary = new Summary();
                session.setAttribute(Constants.SUMMARY_KEY, summary);
            } else
                summary = (Summary) obj;

            summary.getSummary(DBConnections.getDBConnection(request));

            return mapping.findForward("summary");
        }
    }
}
package edu.ucsd.cse.GradAdmissions.Actions;

public class FacultyAction extends Action {

    public ActionForward execute(
            ActionMapping mapping,
            ActionForm form,
            HttpServletRequest request,
            HttpServletResponse response)
            throws Exception {

        HttpSession session;
        Faculty faculty = null;
        FacultyForm facultyForm = null;
        String action;
        boolean result;

        session = request.getSession();
        Faculty faculty = null;
        FacultyForm facultyForm = null;
        String action;
        boolean result;

        session = request.getSession();
        Object obj = session.getAttribute(Constants.FACULTY_KEY);
        if (obj == null || !(obj instanceof Faculty)) {
            faculty = new Faculty();
            session.setAttribute(Constants.FACULTY_KEY, faculty);
        } else {
            faculty = (Faculty) obj;
        }

        facultyForm = (FacultyForm) form;
        action = facultyForm.getAction();

        // default action is view the contents of the table
        if (action == null) {
            faculty.getAllFaculty(DBConnections.getDBConnection(request));
            return mapping.findForward("faculty");
        }

        if (action.equals("Edit")) {
            faculty.getSingleFaculty(
                    DBConnections.getDBConnection(request),
                    facultyForm.getFacultyID());
            // populate form
            facultyForm.setFacultyID(
                    faculty.getFaculty().getRowsByIndex()[0][0].toString());
            facultyForm.setFirstName(
                    faculty.getFaculty().getRowsByIndex()[0][1].toString());
            facultyForm.setLastName(
                    faculty.getFaculty().getRowsByIndex()[0][2].toString());
        }
    }
}
faculty.getFaculty().getRowsByIndex()[0][2].toString();
facultyForm.setEmail(    faculty.getFaculty().getRowsByIndex()[0][3].toString());
facultyForm.setActive(    faculty.getFaculty().getRowsByIndex()[0][4].toString());

return mapping.getInputForward();
}

if (action.equals("Add")) {
    // reset and populate form
    facultyForm.reset(mapping, request);
    facultyForm.action = "Insert";

    return mapping.getInputForward();
}

if (action.equals("Update")) {
    result =
    faculty.updateFaculty(    DBConnections.getDBConnection(request),
    facultyForm.getFirstName(),
    facultyForm.getLastName(),
    facultyForm.getEmail(),
    facultyForm.getActive(),
    facultyForm.getFacultyID());
    faculty.getAllFaculty(DBConnections.getDBConnection(request));
    ActionMessages messages = new ActionMessages();
    ActionMessage message = new
    ActionMessage("message.update.success");
    messages.add(ActionMessages.GLOBAL_MESSAGE, message);
    saveMessages(request, messages);
    return mapping.findForward("faculty");
}

if (action.equals("Insert")) {
    result =
    faculty.addFaculty(    DBConnections.getDBConnection(request),
    facultyForm.getFirstName(),
    facultyForm.getLastName(),
    facultyForm.getEmail(),
    facultyForm.getActive());
    faculty.getAllFaculty(DBConnections.getDBConnection(request));
    ActionMessages messages = new ActionMessages();
    ActionMessage message = new
    ActionMessage("message.insert.success");
    messages.add(ActionMessages.GLOBAL_MESSAGE, message);
    saveMessages(request, messages);
    return mapping.findForward("faculty");
}

if (action.equals("Delete")) {
    result =
    faculty.deleteFaculty(    DBConnections.getDBConnection(request),
    facultyForm.getFacultyID());
    faculty.getAllFaculty(DBConnections.getDBConnection(request));
    ActionMessages messages = new ActionMessages();
    ActionMessage message = new
    ActionMessage("message.delete.success");
    messages.add(ActionMessages.GLOBAL_MESSAGE, message);
    saveMessages(request, messages);
return mapping.findForward("faculty");
}
FacultyForm.java
package edu.ucsd.cse.GradAdmissions.ActionForms;

import java.util.regex.Matcher;
import java.util.regex.Pattern;
import javax.servlet.http.HttpServletRequest;
import org.apache.struts.action.ActionError;
import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionMapping;

public class FacultyForm extends org.apache.struts.action.ActionForm {
    String action;
    String facultyID;
    String firstName;
    String lastName;
    String email;
    String active;

    public String getAction() {
        return action;
    }

    public String getFacultyID() {
        return facultyID;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    public String getEmail() {
        return email;
    }

    public String getActive() {
        return active;
    }

    public void setAction(String string) {
        action = string;
    }

    public void setFacultyID(String string) {
        facultyID = string;
    }

    public void setFirstName(String string) {
        firstName = string;
    }

    public void setLastName(String string) {
        lastName = string;
    }

    public void setEmail(String string) {
        email = string;
    }

    public void setActive(String string) {
if (string == null)
  active = "0";
else
  active = string;
}

public ActionErrors validate(ActionMapping arg0, HttpServletRequest arg1) {

  ActionErrors errors = new ActionErrors();
  Pattern emailPtrn = Pattern.compile("^\S+@\S+$");

  if (action != null
      && action.equals("Update")) {
    if (firstName == null || firstName.length() < 1) {
      errors.add("firstName",
                new ActionError("error.required", "First Name");
    } if (lastName == null || lastName.length() < 1) {
      errors.add("lastName",
                new ActionError("error.required", "Last Name");
    } if (email == null || email.length() < 1) {
      errors.add("email", new ActionError("error.required",
          "Email");
    } else {
      Matcher m = emailPtrn.matcher(email);
      if (!m.matches()) {
        errors.add("email", new ActionError("error.email.format", "Email");
      }
    }
  }

  return errors;
}

public void reset(ActionMapping arg0, HttpServletRequest arg1) {
  this.active = "0";
  this.firstName = null;
  this.lastName = null;
  this.email = null;
  this.facultyID = null;
  super.reset(arg0, arg1);
}
<%@ page language="java" import="java.sql.*" session="true" %>
<%@ taglib uri="struts/bean-el" prefix="bean" %>
<%@ taglib uri="struts/html-el" prefix="html" %>
<%@ taglib uri="struts/logic-el" prefix="logic" %>
<%@ taglib uri="jstl/c" prefix="c" %>

<html:html locale="true">
<head>
<title><bean:message key="index.title" /></title>
<jsp:directive.include file="/header_head.jsp" />
<html:base />
</head>
<body>

<jsp:include page="/header_body.jsp" flush="true" />
<jsp:include page="/header_inner.jsp" flush="true">
<jsp:param name="pageTitle" value="all.menu.Summary" />
</jsp:include>

<logic:messagesPresent message="true">
<html:messages id="message" message="true">
<p class="infomessage">
> <c:out value="${message}" escapeXml="false" />
</p>
</html:messages>
</logic:messagesPresent>

<logic:messagesPresent>
<html:messages id="error">
<p class="errormessage">
> <c:out value="${error}" escapeXml="false" />
</p>
</html:messages>
</logic:messagesPresent>

<table cellpadding="2" cellspacing="1" border="0" bgcolor="#999999" width="512">
<tr>
<td bgcolor="#FFFFFF">
<table border="0" cellpadding="2" cellspacing="1" width="100%">
<tr>
<td class="listheader"><img border="0" width="1" height="1" src="<%= request.getContextPath() %>/images/space.gif" /></td>
<td class="assigngreen"><b>Domestic</b></td>
<td class="assignred"><b>Foreign</b></td>
</tr>
<tr class="listline">
<td class="listheader"><b>Incomplete</b></td>
<c:forEach var="row" items="${summaryBean.summary.rowsByIndex}" begin="0" end="1">
<td class="body2"><b><c:out value="${row[0]}" /></b></td>
</c:forEach>
</tr>
<tr class="listline">
<td class="listheader"><b>Under Review</b></td>
<c:forEach var="row" items="${summaryBean.summary.rowsByIndex}" begin="2" end="3">
<td class="body2"><b><c:out value="${row[0]}" /></b></td>
</c:forEach>
</tr>
</table>
</td>
</tr>
</table>
</jsp:include>
</jsp:include>
<table>
<thead>
<tr>
<th>Category</th>
<th>Number of Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rejected</td>
<td></td>
</tr>
<tr>
<td>Admittable</td>
<td></td>
</tr>
<tr>
<td>Admitted</td>
<td></td>
</tr>
<tr>
<td>Accepted</td>
<td></td>
</tr>
<tr>
<td>Denied</td>
<td></td>
</tr>
</tbody>
</table>
Add or update faculty member.
**faculty.jsp**

```html
<%@ page language="java" import="java.sql.*" session="true" %>
<%@ taglib uri="struts/bean-el" prefix="bean" %>
<%@ taglib uri="struts/html-el" prefix="html" %>
<%@ taglib uri="struts/logic-el" prefix="logic" %>
<%@ taglib uri="jstl/c" prefix="c" %>

<html:html locale="true">
    <head>
        <title>${messageBean.message["index.title"]}</title>
        <jsp:directive.include file="/header_head.jsp" />
        <html:base />
    </head>

    <body>
        <jsp:include page="/header_body.jsp" flush="true" />
        <jsp:include page="/header_inner.jsp" flush="true">
            <jsp:param name="pageTitle" value="admin.menu.Faculty" />
        </jsp:include>
        <p>
            <font class="body1" size="-1" face="verdana,arial, geneva, sans-serif">
                Add, remove, or edit faculty members.
            </font>
        </p>

        <logic:messagesPresent message="true">
            <html:messages id="message" message="true">
                <p class="infomessage">
                    <c:out value="${message}" escapeXml="false" />
                </p>
            </html:messages>
        </logic:messagesPresent>

        <logic:messagesPresent>
            <html:messages id="error" message="true">
                <p class="errormessage">
                    <c:out value="${error}" escapeXml="false" />
                </p>
            </html:messages>
        </logic:messagesPresent>

        <table cellpadding="2" cellspacing="1" border="0" bgcolor="#999999" width="512">
            <tr>
                <td bgcolor="#FFFFFF">
                    <table border="0" cellpadding="2" cellspacing="1" width="100%">
                        <tr>
                            <td class="listheader">
                                Name
                            </td>
                            <td class="listheader">
                                <c:out value="${facultyBean.faculty.columnNames[3]}" />
                            </td>
                            <td class="listheader">
                                <c:out value="${facultyBean.faculty.columnNames[4]}" />
                            </td>
                            <td class="listheader">
                                Actions
                            </td>
                        </tr>
                        <tr>
                            <td class="listheader">
                                Name
                            </td>
                        </tr>
                    </table>
                </td>
            </tr>
        </table>
    </body>
</html:html>
```
<table>
<thead>
<tr>
<th>Column 1</th>
<th>Column 2</th>
<th>Column 3</th>
<th>Column 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;b&gt;Row 1&lt;/b&gt;, Column 1</td>
<td>Mailto:Row2</td>
<td>Yes</td>
<td>&lt;form action=&quot;/admin/faculty&quot; method=&quot;POST&quot; class=&quot;body2&quot;&gt;&lt;input type=&quot;hidden&quot; value=&quot;Add&quot; name=&quot;action&quot; /&gt; &lt;input type=&quot;submit&quot; class=&quot;body2&quot; value=&quot;Add&quot; /&gt; &lt;/form&gt;</td>
</tr>
<tr>
<td>&lt;b&gt;Row 2&lt;/b&gt;, Column 1</td>
<td>Mailto:Row3</td>
<td>No</td>
<td>&lt;form action=&quot;/admin/faculty&quot; method=&quot;POST&quot; class=&quot;body2&quot;&gt;&lt;input type=&quot;hidden&quot; value=&quot;Edit&quot; name=&quot;action&quot; /&gt; &lt;input type=&quot;hidden&quot; value=&quot;Row0&quot; name=&quot;facultyID&quot; /&gt; &lt;input type=&quot;submit&quot; class=&quot;body2&quot; value=&quot;Update&quot; /&gt; &lt;/form&gt;</td>
</tr>
<tr>
<td>&lt;b&gt;Row 3&lt;/b&gt;, Column 1</td>
<td>Mailto:Row4</td>
<td>Yes</td>
<td>&lt;form action=&quot;/admin/faculty&quot; method=&quot;POST&quot; class=&quot;body2&quot;&gt;&lt;input type=&quot;hidden&quot; value=&quot;Delete&quot; name=&quot;action&quot; /&gt; &lt;input type=&quot;hidden&quot; value=&quot;Row0&quot; name=&quot;facultyID&quot; /&gt; &lt;input type=&quot;submit&quot; class=&quot;body2&quot; value=&quot;Delete&quot; /&gt; &lt;/form&gt;</td>
</tr>
</tbody>
</table>
package edu.ucsd.cse.GradAdmissions.Beans;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import edu.ucsd.cse.GradAdmissions.DatabaseException;

public class State {
    String currentYear;
    String userID;
    String userRole;
    String userName;

    public String getCurrentYear() {
        return currentYear;
    }

    public void setCurrentYear(String string) {
        currentYear = string;
    }

    public String getUserID() {
        return userID;
    }

    public String getUserRole() {
        return userRole;
    }

    public String getUserName() {
        return userName;
    }

    public boolean getCurrentYear(Connection conn) throws DatabaseException {
        String sqlString = null;
        Statement stmt = null;
        ResultSet resultSet = null;

        try {
            sqlString = "SELECT AdmissionsYear FROM AdmissionsYears WHERE isCurrent = 1";
            stmt = conn.createStatement();
            resultSet = stmt.executeQuery(sqlString);
            resultSet.next();
            currentYear = resultSet.getString(1);
        }
        catch (SQLException e) {
            throw new DatabaseException(e.getMessage());
        }
        finally {
            try {
                if (conn != null)
                    conn.close();
                if (stmt != null)
stmt.close();
}

try {
  if (userRoleStr.equals("Chair")
      || userRoleStr.equals("Committee")
      || userRoleStr.equals("Faculty"))
    sqlString = "SELECT FacultyID, FirstName, LastName FROM Faculty WHERE Email LIKE ?";
  else if (userRoleStr.equals("Admin") || userRoleStr.equals("Staff"))
    sqlString = "SELECT StaffID, FirstName, LastName FROM Staff WHERE Email LIKE ?";
  pstmt = conn.prepareStatement(sqlString);
  pstmt.setString(1, userLoginStr + "");
  resultSet = pstmt.executeQuery();
  resultSet.next();
  userID = resultSet.getString(1);
  userName = resultSet.getString(2) + " " + resultSet.getString(3);
}

} catch (SQLException e) {
  throw new DatabaseException(e.getMessage());
}

finally {
  try {
    if (conn != null)
      conn.close();
    if (pstmt != null)
      pstmt.close();
  } catch (SQLException sqle) {
    throw new DatabaseException(sqle.getMessage());
  }
}

return true;
}
User: Andrew Kovacevic (Admin)

<table>
<thead>
<tr>
<th>State of Review Process</th>
<th>Domestic</th>
<th>Foreign</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incomplete</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Under Review</td>
<td>479</td>
<td>1069</td>
</tr>
<tr>
<td>Rejected</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Admittable</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Admitted</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Accepted</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Denied</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

9500 Gilman Drive, La Jolla, CA 92030-0114

Feedback

Official web page of the University of California, San Diego

Copyright © 2004 Regents of the University of California. All rights reserved.
Manage Faculty Members
Add, remove, or edit faculty members.

<table>
<thead>
<tr>
<th>Name</th>
<th>Email</th>
<th>Active Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baden, Scott</td>
<td><a href="mailto:baden@cs.ucsd.edu">baden@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Bafna, Vineet</td>
<td><a href="mailto:vbafna@cs.ucsd.edu">vbafna@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>bar, foo</td>
<td>foo@bar</td>
<td>Yes</td>
</tr>
<tr>
<td>Bellare, Mihir</td>
<td><a href="mailto:mihir@cs.ucsd.edu">mihir@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Belongie, Serge</td>
<td><a href="mailto:sjb@cs.ucsd.edu">sjb@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Berman, Francine</td>
<td><a href="mailto:berman@cs.ucsd.edu">berman@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Burkhard, Walter</td>
<td><a href="mailto:burkhard@cs.ucsd.edu">burkhard@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Calder, Brad</td>
<td><a href="mailto:calder@cs.ucsd.edu">calder@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Carter, Larry</td>
<td><a href="mailto:carter@cs.ucsd.edu">carter@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Cheng, Chung-Kuan</td>
<td><a href="mailto:kuan@cs.ucsd.edu">kuan@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Chien, Andrew</td>
<td><a href="mailto:achien@cs.ucsd.edu">achien@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Chung Graham, Fan</td>
<td><a href="mailto:fan@ucsd.edu">fan@ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Cottrell, Garrison</td>
<td><a href="mailto:gary@ucsd.edu">gary@ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Dasgupta, Sanjoy</td>
<td><a href="mailto:dasgupta@cs.ucsd.edu">dasgupta@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Deutsch, Alin</td>
<td><a href="mailto:deutsch@cs.ucsd.edu">deutsch@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Elkan, Charles</td>
<td><a href="mailto:elkan@cs.ucsd.edu">elkan@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Eskin, Eleazar</td>
<td><a href="mailto:eeskin@cs.ucsd.edu">eeskin@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Ferrante, Jeanne</td>
<td><a href="mailto:ferrante@cs.ucsd.edu">ferrante@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Goguen, Joseph</td>
<td><a href="mailto:goguen@cs.ucsd.edu">goguen@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Graham, Ronald</td>
<td><a href="mailto:rgraham@cs.ucsd.edu">rgraham@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Name</td>
<td>Email</td>
<td>Role</td>
</tr>
<tr>
<td>-------------------</td>
<td>---------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Griswold, William</td>
<td><a href="mailto:wgg@cs.ucsd.edu">wgg@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Gupta, Rajesh</td>
<td><a href="mailto:gupta@cs.ucsd.edu">gupta@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Howden, William</td>
<td><a href="mailto:howden@cs.ucsd.edu">howden@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Hu, Te C.</td>
<td><a href="mailto:hu@cs.ucsd.edu">hu@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Impagliazzo, Russell</td>
<td><a href="mailto:russell@cs.ucsd.edu">russell@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Jensen, Henrik Wann</td>
<td><a href="mailto:henrik@cs.ucsd.edu">henrik@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Kahng, Andrew</td>
<td><a href="mailto:abk@cs.ucsd.edu">abk@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Karin, Sidney</td>
<td><a href="mailto:karin@cs.ucsd.edu">karin@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Kriegman, David</td>
<td><a href="mailto:kriegman@cs.ucsd.edu">kriegman@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Krueger, Ingolf</td>
<td><a href="mailto:ikrueger@cs.ucsd.edu">ikrueger@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Marzullo, Keith</td>
<td><a href="mailto:marzullo@cs.ucsd.edu">marzullo@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Micciancio, Daniele</td>
<td><a href="mailto:danielle@cs.ucsd.edu">danielle@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Orailoglu, Alex</td>
<td><a href="mailto:alex@cs.ucsd.edu">alex@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Papakonstantinou, Yannis</td>
<td><a href="mailto:vannis@cs.ucsd.edu">vannis@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Pasquale, Joseph</td>
<td><a href="mailto:pasquale@cs.ucsd.edu">pasquale@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Pevzner, Pavel</td>
<td><a href="mailto:ppevzner@cs.ucsd.edu">ppevzner@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Ramamohan, Paturi</td>
<td><a href="mailto:paturi@cs.ucsd.edu">paturi@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Rangan, Venkat</td>
<td><a href="mailto:venkat@cs.ucsd.edu">venkat@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Savage, Stefan</td>
<td><a href="mailto:ssavage@cs.ucsd.edu">ssavage@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Smarr, Larry</td>
<td><a href="mailto:lsmarr@cs.ucsd.edu">lsmarr@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Snoeren, Alex</td>
<td><a href="mailto:snoeren@cs.ucsd.edu">snoeren@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Tullsen, Dean</td>
<td><a href="mailto:tullsen@cs.ucsd.edu">tullsen@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Vahdat, Amin</td>
<td><a href="mailto:vahdat@cs.ucsd.edu">vahdat@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Varghese, George</td>
<td><a href="mailto:varghese@cs.ucsd.edu">varghese@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Vianu, Victor</td>
<td><a href="mailto:vianu@cs.ucsd.edu">vianu@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Voelker, Geoff</td>
<td><a href="mailto:gvoelker@cs.ucsd.edu">gvoelker@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
<tr>
<td>Williamson, Gill S.</td>
<td><a href="mailto:gill@cs.ucsd.edu">gill@cs.ucsd.edu</a></td>
<td>Yes</td>
</tr>
</tbody>
</table>
## Manage Faculty Members

Add or update faculty member.

<table>
<thead>
<tr>
<th>First Name:</th>
<th>Scott</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Name:</td>
<td>Baden</td>
</tr>
<tr>
<td>Email:</td>
<td><a href="mailto:baden@cs.ucsd.edu">baden@cs.ucsd.edu</a></td>
</tr>
<tr>
<td>Active:</td>
<td>✔</td>
</tr>
</tbody>
</table>

Update  | Reset

9500 Gilman Drive, La Jolla, CA 92093-0114
Manage Faculty Members
Add, remove, or edit faculty members.

> Successful deletion.

<table>
<thead>
<tr>
<th>Name</th>
<th>Email</th>
<th>Active</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baden, Scott</td>
<td><a href="mailto:baden@cs.ucsd.edu">baden@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Bafna, Vineet</td>
<td><a href="mailto:vbafna@cs.ucsd.edu">vbafna@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Bellare, Mihir</td>
<td><a href="mailto:mihir@cs.ucsd.edu">mihir@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Belongie, Serge</td>
<td><a href="mailto:sjb@cs.ucsd.edu">sjb@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Berman, Francine</td>
<td><a href="mailto:berman@cs.ucsd.edu">berman@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Burkhard, Walter</td>
<td><a href="mailto:burkhard@cs.ucsd.edu">burkhard@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Calder, Brad</td>
<td><a href="mailto:calder@cs.ucsd.edu">calder@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Carter, Larry</td>
<td><a href="mailto:carter@cs.ucsd.edu">carter@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Cheng, Chung-Kuan</td>
<td><a href="mailto:kuan@cs.ucsd.edu">kuan@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Chien, Andrew</td>
<td><a href="mailto:achien@cs.ucsd.edu">achien@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Chung Graham, Fan</td>
<td><a href="mailto:fan@ucsd.edu">fan@ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Cottrell, Garrison</td>
<td><a href="mailto:gary@ucsd.edu">gary@ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Dasgupta, Sanjoy</td>
<td><a href="mailto:dasgupta@cs.ucsd.edu">dasgupta@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Deutsch, Alin</td>
<td><a href="mailto:deutsch@cs.ucsd.edu">deutsch@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Elkan, Charles</td>
<td><a href="mailto:elkan@cs.ucsd.edu">elkan@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Eskin, Eleazar</td>
<td><a href="mailto:eeskin@cs.ucsd.edu">eeskin@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Ferrante, Jeanne</td>
<td><a href="mailto:ferrante@cs.ucsd.edu">ferrante@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Goguen, Joseph</td>
<td><a href="mailto:goguen@cs.ucsd.edu">goguen@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
<tr>
<td>Graham, Ronald</td>
<td><a href="mailto:rgraham@cs.ucsd.edu">rgraham@cs.ucsd.edu</a></td>
<td>Yes</td>
<td>Edit</td>
</tr>
</tbody>
</table>