

Chair of Governors Training Tracker

CG4.4 - Software Maintenance

John Robinson

09/01/2014

Contents

Subroutines	2
Use of the User Confirmation Box	2
How the Join Table Function is used.....	5
Dates In and Out of the Database.....	8
Searching the Database with an ID	9
Forms	10
ER Diagram and Database.....	11
Governor Table	11
Course Taken Table.....	11
Course Table	11
Category Table	12
Variables	13
Add Category.....	13
Add Completed Course	13
Add Course.....	14
Add Governor.....	14
Edit Category.....	14
Edit Course	15
Edit Governor.....	15
Remove Category.....	16
Remove Completed Course	16
Remove Course	17
Remove Governor.....	17
User Confirmation Box.....	17
Validation.....	17
View Categories	18
View Completed Courses by Course	18
View Completed Courses by Governor.....	18
View Courses.....	19
View Governors.....	19

Software Maintenance

Subroutines

Use of the User Confirmation Box

The UserConfirmationBox class is usually used when you want to alert the user of an action they are about to perform such as removing something from the database. An example of this can be found in the RemoveCategory class.

Below is a screenshot of the code surrounding the activation of a confirmation box:

```
myCat = categoryComboBox.getSelectedItem().toString(); //get category info
if (myCat.equals("-")) { //check to make sure the user has selected a category
    removeCategory = false;
    //show an error to the user
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("Please select a category to remove.");
    MyMessage.setVisible(true);
} else { //if there's no error
    UserConfirmationBox userConfirm = new UserConfirmationBox(this, true);
    //make sure the user knows what they're removing
    userConfirm.setMessage("Removing Category:\n"
        + "The category that you wish to remove is:\n"
        + myCat
        + ".\nUpon deleting this category,\nany course or completed"
        + "\ncourses that require this category\nwill also be"
        + "deleted.\nAre you sure you want to continue?");
    userConfirm.setVisible(true);
    removeCategory = userConfirm.removeConfirmation(); //take the result from the
    userConfirm.dispose(); //make sure we close it
```

I'll use the example in the RemoveCategory class.

Above is the beginning of the method "removeCategoryBtnActionPerformed" which activates when the Remove Category button is pressed on the form. After checking to make sure a category has been selected, we can make sure they want to remove the selected category. In the else statement, we make a new UserConfirmationBox object to use as our buffer.

We use the parent as "this" meaning the object it was created in will be the parent. This is because the confirmation box will lock the user out of being able to edit the parent windows of the confirmation box.

After creating the object, we set the message. Place here all the appropriate information that is required to help the user make their decision. Here I have explained that they will be removing the category selected and any records that link to it.

After the message is set, make the window visible to the user and prevent them from editing parent windows.

```
*/  
private void yesBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    removeConfirmation = true;  
    setVisible(false);  
}
```

There are only 2 buttons on the user confirmation window. A Yes and a No button. If the yes button is pressed, a variable is set to true for use in a moment. If No, then:

```
/* allow the you to retrieve the variable before closing this object  
*/  
private void noBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    removeConfirmation = false;  
    setVisible(false);  
}
```

Set the same variable to false. In both situations, hide the window. This puts the previous method back in action. Here it is again:

```
        + "deleted.\nAre you sure you want to continue?");  
userConfirm.setVisible(true);  
removeCategory = userConfirm.removeConfirmation; //take the result f  
userConfirm.dispose(); //make sure we close it  
if (removeCategory == false) { //if they don't want to delete  
    UserMessageBox MyMessage = new UserMessageBox();  
    MyMessage.setMessage("The Category:\n"  
        + myCat  
        + ".\nWas not removed.");  
    MyMessage.setVisible(true);  
}
```

Here we see that after the confirmation box is set visible, we wait until it's closed before continuing. Once the user has selected an option we retrieve the variable from the object and store it for use later. Now we make sure we close the object otherwise if we try to run this method again it may cause errors.

After this, we can use the final confirmation value to decide to remove the selected category. This is done with a simple IF statement. Above is the result if the value was set to false. Below is the result if set to true.

```

if (removeCategory == true) { //if they want to delete
    myCatArray = myCat.split(","); //split the category information
    //make a string to delete the category by the given ID
    myLocalString = "delete from category where CatID=" + myCatArray[0];
    CG41App.dbObject.sqlString = myLocalString; //set the query
    thisError = CG41App.dbObject.deleteRecord(); //execute

    //reset the UI for the user to remove another category
    categoryCombox.removeAllItems();
    categoryCombox.addItem("-");
    categoryCombox.setSelectedIndex(0);
    loadCategories();

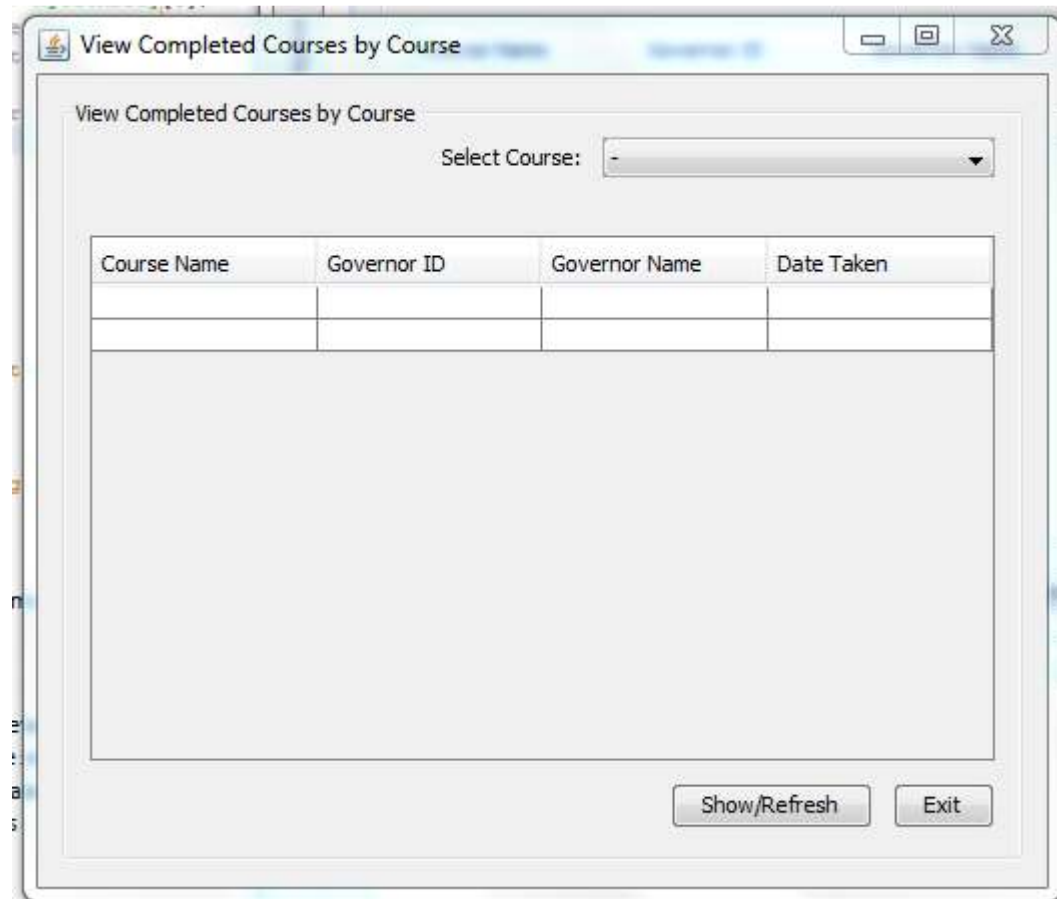
    // Report completion to the user
    if (thisError == 0) { //if there's no error
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("Category Successfully Deleted");
        MyMessage.setVisible(true);
    } else { //if there's an error
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("There was an Error deleting the Category.");
        MyMessage.setVisible(true);
    }
}

```

We can see that the record is removed from the database if the user confirms that it is what they want to happen.

How the Join Table Function is used

Table Joins are a function of MySQL. They allow you to retrieve data from several fields in different tables at the same time. This prevents you having to write several database searches to get all the information you need. This is particularly useful if you have a relational database and you want to display information on the related data. Below is an example from the class “ViewCompletedCoursesByCourse”.



The screenshot shows a Java Swing window titled "View Completed Courses by Course". Inside the window, there is a label "View Completed Courses by Course" and a "Select Course:" dropdown menu. Below the dropdown is a table with four columns: "Course Name", "Governor ID", "Governor Name", and "Date Taken". The table is currently empty. At the bottom right of the window, there are two buttons: "Show/Refresh" and "Exit".

Above is a screenshot of the form from the class. In the top right there is a combo box which will allow the user to select a course. Then we have a table to display all the people that have taken this course. But, because there are no governor records in the completed course table, we need to use the governor ID from that table to find the related governor record in the governor table.

So the user selects a course then pressed the Show/Refresh button.

Then we can begin to populate the table. As with populating other GUI components such as combo boxes and other tables, we need to count the amount of records in the coursetaken table where the selected course ID matches the one in the database.

All the following code can be found in the method “refreshBtnActionPerformed”.

```

myCourseArray = myCourse.split(","); //get the ID out
myCourseID = Integer.parseInt(myCourseArray[0]); //store the ID

//count the amount of records where the selected ID is concerned
myLocalString = "select COUNT(*) from coursetaken where CourseID = "
                + myCourseID;
CG41App.dbObject.sqlString = myLocalString;
thisError = CG41App.dbObject.getCountBySelect();//execute

numRecords = CG41App.dbObject.NumberOfRecords; //save number of records

```

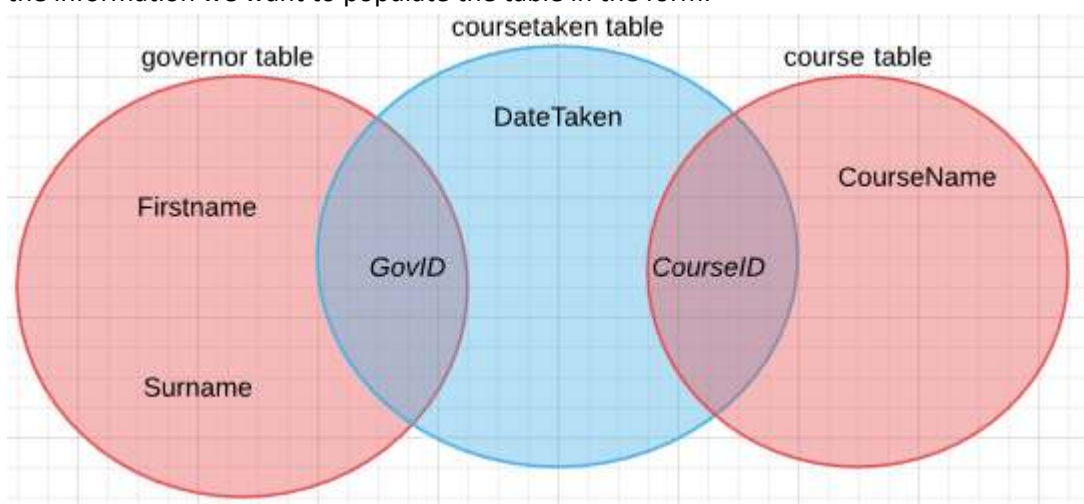
Above we take the course ID from the combo box and then use to search the coursetaken table to find how many records there are that match the course IDs. We can use this to make sure we have the right amount of columns in the table.

```

// Construct an SQL statment to find all the information we need:
//The information that we're pulling here is the course name, datetaken,
//Governor ID and Governor First/Last names but only where the course ID
//selected and coursetaken course ID match.
//The joins allow us to pull the information from the three tables
myLocalString = "SELECT course.CourseName, coursetaken.DateTaken, "
                + "coursetaken.GovID, governor.Firstname, governor.Surname "
                + "FROM (coursetaken "
                + "INNER JOIN course ON course.CourseID = coursetaken.CourseID) "
                + "INNER JOIN governor ON governor.GovID = coursetaken.GovID "
                + "WHERE (coursetaken.CourseID = " + myCourseID + ")";
CG41App.dbObject.sqlString = myLocalString;

```

This piece of code is the actual JOIN statement. This is where we can combine the tables to retrieve the information we want to populate the table in the form.



Above is a diagram of how the tables are related. Each item in a circle represents a field we need to use for the JOIN statement to get all the relevant information.

First, we SELECT what fields we want to take the data from. Just make sure they're prefixed with what table in the database they are found in. Then, we say where we want to select the information from and how they're related.

So; *FROM (coursetaken INNER JOIN course ON course.CourseID = coursetaken.CourseID)*

FROM just states where the middle table is residing and where to pull the information from. FROM coursetaken just means take FROM the coursetaken table. But we need to link tables together, so INNER JOIN signifies a JOIN in the tables and INNER is just the type of JOIN. So we're JOINing the table course ON the condition that course.CourseID (the field CourseID from the course table) = coursetaken.CourseID (the field CourseID in the coursetaken table). This is setting up the foreign key reference between the tables and joining them.

The next part: *INNER JOIN governor ON governor.GovID = coursetaken.GovID* is more of the same. We're joining the tables "governor" and "coursetaken" by the field "GovID".

The final part is the WHERE statement. This is the part which allows us to only select certain records. We want the records where the course ID in the coursetaken table matches the selected course ID. So: *WHERE (coursetaken.CourseID = "+myCourseID + ")* Allows us to do just that.

Now, in our RecordSet in the database object, we have all the fields from the select statement (same as the Venn diagram) populated with data (as long as there's entries for completed courses with the selected CourseID).

Now we can move on and place it all into the table in the form.

```
thisError = CG41App.dbObject.getRecordSetBySelect();
if (thisError == 0) {
    // No error on execution of statement
    // In order to make a table, you need to create an array of objects
    // It must be as long as the number of records you've counted and
    // must be as wide as the number of columns you've asked for
    Object[][] myTableData = new Object[numRecords][4];
    String[] myColumnNames = {"Course Name", "Governor ID",
        "Governor Name", "Date Taken"};
    .
```

Create a new table object and give it column names for the data to be displayed.

```
try {
    int row = 0;
    while (CG41App.dbObject.rs.next()) { // For each record
        //get each part of the record from the record set
        courseName = CG41App.dbObject.rs.getString("CourseName");
        govID = CG41App.dbObject.rs.getString("GovID");
        //concatenate the governor name
        govName = CG41App.dbObject.rs.getString("Firstname") + " "
            + CG41App.dbObject.rs.getString("Surname");
        dateTaken = CG41App.dbObject.rs.getDate("DateTaken");
        myTableData[row][0] = courseName;
        myTableData[row][1] = govID;
        myTableData[row][2] = govName;
        myTableData[row][3] = String.valueOf(dateTaken);
        row = row + 1;           // Next row in table
    }
    .
```

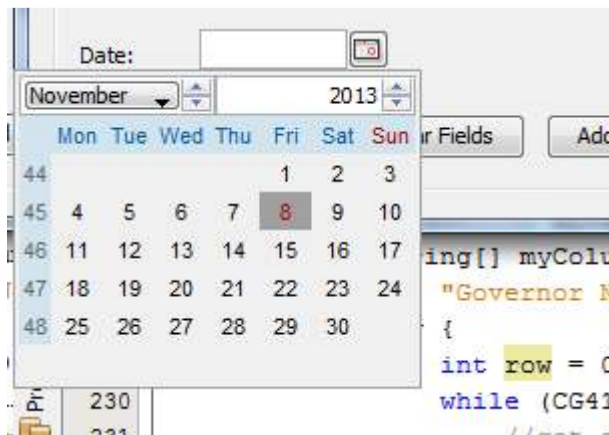

Now we just have to take the data from the record set and place it into the rows in the table. For each field, we have to retrieve the data by the field name in the database.

```
// Have the table data, so make a new table with it
JTable jTableCoursesByID = new JTable(myTableData, myColumnNames);
jTableCoursesByID.setFillsViewportHeight(true);
courseScrlPane.getViewport().add(jTableCoursesByID);
courseScrlPane.repaint();
```

The above image just displays the table to the user.

Dates In and Out of the Database

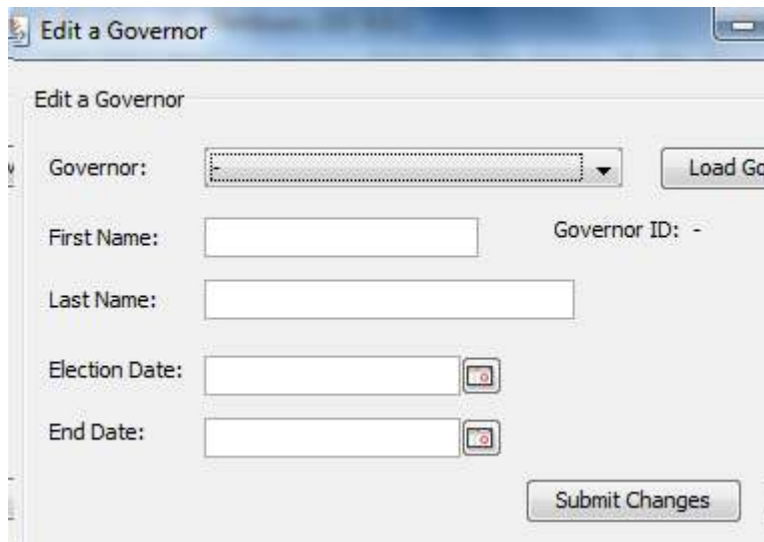
To make sure we don't encounter any errors trying to place or retrieve dates from the database, we need to format any dates the user defines. I have used a third party library which gave me access to a GUI calendar to make this process a bit easier. This prevents the user from entering a date into the system that doesn't exist; the 31st November for example. I did not write any of the code in this library.



Above is an example of the GUI component I have used.

```
courseDate = compCourseDateChooser.getDate(); //get date and format
courseCompDateString = String.format("%1$tY-%1$tm-%1$td", courseDate);
```

Above is how I retrieve the dates and then format them for the database. The format function uses standard Regex and the documentation can be found in the official Java Docs. But for this example, I have formatted it year first, then month, then day. All separated by a dash. This is a standard way of formatting it for MySQL databases. I have chosen to turn the Date data type from the date chooser into a string for easier adding to the database.



On retrieving dates from the database to be used in an edit form, for example, have to be set in the date choosers. This is easier than setting in the database because once it has been pulled from the database it can just be set as the date chooser will accept a Date data type from the database in its raw form.

```
elecDateChooser.setDate(CG41App.dbObject.rs.getDate("ElectionDate"));  
endDateChooser.setDate(CG41App.dbObject.rs.getDate("EndDate"));
```

Above is the code to put it into the date chooser.

Searching the Database with an ID

There are a few occasions where the user will select something from a combo box and the ID of the selected item will have to be used to search the database to find more data.

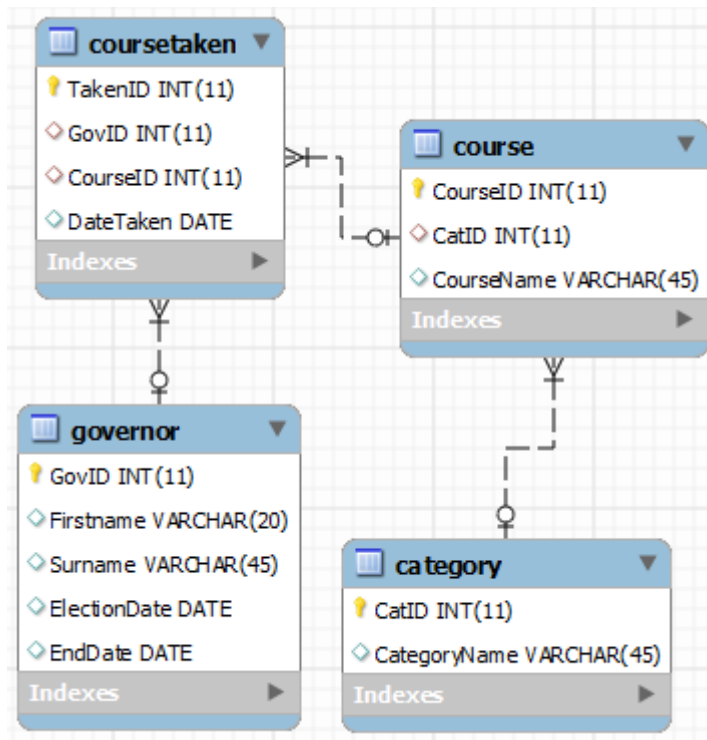
Where this happens, the selected item from a combo box will be taken and split into an array. The first item in the array will be the ID. The ID can then be used in a MySQL query to find the record it belongs to. There are examples of code that follows this in the edit classes and in the remove classes.

Forms

Below is a list of all the classes in my program that contain forms. These classes make up my GUI. I have not included any classes that I did not write myself.

Name of Class	Description
AddCategory	Adds a category to the database.
AddCompletedCourse	Adds a completed course to the database.
AddCourse	Adds a course to the database.
AddGovernor	Adds a governor to the database.
CG41View	Main application window.
EditCategory	Edits a category already in the database.
EditCourse	Edits a course already in the database.
EditGovernor	Edits a governor already in the database.
RemoveCategory	Removes a category already in the database.
RemoveCompletedCourse	Removes a completed course already in the database.
RemoveCourse	Removes a course already in the database.
RemoveGovernor	Removes a governor already in the database.
UserConfirmationBox	Used as a buffer to make sure the user knows what action(s) they are taking.
ViewCategories	Displays a list of categories in the database.
ViewCompletedCoursesByCourse	Displays a list of the completed courses but only of the course selected by the user.
ViewCompletedCoursesByGovernor	Displays a list of the completed courses but only by the governor selected by the user.
ViewCourses	Displays a list of courses in the database.
ViewGovernors	Displays a list of governors in the database.

ER Diagram and Database



Above we have an ER diagram of my database design. Below are explanations of the tables.

Governor Table

Name	Type	Example	On Delete Action
GovID	Auto Int	2	None
Firstname	String	James	None
Surname	String	Robinson	None
ElectionDate	Date	2013-12-12	None
EndDate	Date	2014-05-28	None

governor: **GovID**, Firstname, Surname, ElectionDate, EndDate.

Course Taken Table

Name	Type	Example	On Delete Action
TakenID	Auto Int	2	None
<i>GovID</i>	Auto Int	5	Cascade Delete Records
<i>CourseID</i>	Auto Int	4	Cascade Delete Records
DateTaken	Date	2013-12-12	None

coursetaken: **TakenID**, *GovID*, *CourseID*, DateTaken.

Course Table

Name	Type	Example	On Delete Action
CourseID	Auto Int	2	None
<i>CatID</i>	Auto Int	7	Cascade Delete Records
CourseName	String	Legal	None

course: **CourseID**, *CatID*, CourseName.

Category Table

Name	Type	Example	On Delete Action
CatID	Auto Int	2	None
CategoryName	Auto Int	7	Cascade Delete Records

category: **CatID**, CourseName.

Variables

Here all the variables will be listed by class in the order that they appear in my IDE. I will not be documenting variables in the classes that are present in my project but did not create myself.

Add Category

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	errorMessage	Used to store a local error message for display to the user. For instance, if something doesn't pass validation.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.

Add Completed Course

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	governor	Used to store the concatenated information pulled from the database on the governor before adding it to the combo box.
String	courseCompDateString	Used to store the formatted string version of the selected date.
String	course	Stores the course name and ID from the database before adding it to the course combo box.
String	compGov	Stores the selected governor from the combo box.
String	completedCourse	Stores the selected course from the combo box.
String	compCourseIDArray[]	Stores the split completedCourse string to retrieve the course ID.
String	compGovIDArray[]	Stores the split compGov string to retrieve the course ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	courseID	Stores the course ID from the split array.
Integer	govID	Stores the Governor ID from the split array.
Integer	i	Used for iteration in for loops.
Date	courseDate	Used to store the date from the date chooser before processing.

Add Course

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	catSelect	Stores the category selected from the combo box.
String	errorMessage	Used to store a local error message for display to the user. For instance, if something doesn't pass validation.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	i	Used for iteration in for loops.
Integer	catIDSelect	Stores the category ID found in the database.

Add Governor

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	endDateString	The formatted string version of the end of term date.
String	elecDateString	The formatted string version of the election date.
String	errorMessage	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Date	endDate	Stores the date from the end of term date chooser.
Date	elecDate	Stores the date from the start of term date chooser.

Edit Category

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	category	Stores the concatenated string which contains the category name and ID to add to the combo box.
String	myCat	Used to store the selected category to then split.
String	errorMessage	Used to store a local error message for display to the user. For instance, if something doesn't pass validation.
String	myCatArray[]	Used as temp storage for the split selected category to get the category ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.

Edit Course

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	myLocalString2	Used to store the MySQL string query locally before set in the database class. This is for the second database object to make sure the two don't collide.
String	course	Stores the concatenated course name and ID to add it to the combo box.
String	myCourse	Stores the selected course from the combo box.
String	catName	Used as a return value to get the category name from a category ID.
String	errorMessage	Used to store a local error message for display to the user. For instance, if something doesn't pass validation.
String	category	Stores the concatenated category name and ID to add it to the combo box.
String	myCat	Stores the selected course from the combo box.
String	myCategoryArray[]	Used as temp storage for the split selected category to get the category ID.
String	myCourseArray[]	Used as temp storage for the split selected course to get the course ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	thisError2	Used to store the error code retrieved from the database query if something goes wrong on execution of a query. This is used for the second database object to make sure the two don't collide.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.

Edit Governor

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	endDateString	The formatted string version of the end of term date.
String	elecDateString	The formatted string version of the election date.
String	errorMessage	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
String	governor	Stores the concatenated governor name and ID to add it to the combo box.
String	myGov	Stores the selected governor from the combo box for processing.
String	myGovArray[]	Used to store the split of myGov to obtain the governor ID.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Date	endDate	Stores the date from the end of term date chooser.
Date	elecDate	Stores the date from the start of term date chooser.

Remove Category

Type	Name	Description
Boolean	removeCategory	Stores confirmation on the removal in progress to prevent it continuing if there's an error or if the user cancels. Similar to myLocalError.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	category	Stores the concatenated string which contains the category name and ID to add to the combo box.
String	myCat	Used to store the selected category to then split.
String	errorMessage	Used to store a local error message for display to the user. For instance, if something doesn't pass validation.
String	myCatArray[]	Used as temp storage for the split selected category to get the category ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.

Remove Completed Course

Type	Name	Description
Boolean	myLocalError	This variable is used to store the current state of errors in this class.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	courseName	Holds the course name in the method getCourseName.
String	myGov	Stores the governor selected from the combo box.
String	myCompCourse	Stores the completed course selected from the combo box.
String	governor	Used to hold a concatenated governor record containing governor name, governor ID.
String	myGovArray[]	Used to hold myGov after it has been split to get the governor ID.
String	myCompCourseArray[]	Used to hold myCompCourse after it has been split to get the completed course ID.
String	compCourseEntry	Stores the concatenated record for each completed course before adding it to the combo box.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	courseID	Stores the course ID from the database.
Integer	thisError2	Similar to thisError, but for the second database object to make sure the two don't collide.

Remove Course

Type	Name	Description
Boolean	removeCourse	Stores confirmation on the removal in progress to prevent it continuing if there's an error or if the user cancels. Similar to myLocalError.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	myLocalString2	Used to store the MySQL string query locally before set in the database class. This one is used for a second DB object. This ensures the two don't collide.
String	catName	Used as a return value when getting the category name by a given ID.
String	myCourse	Stores the course selected from the combo box.
String	course	Used to hold a concatenated course record containing course name, category name and course ID.
String	myCourseArray[]	Used to hold myCourse after it has been split to get the course ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.

Remove Governor

Type	Name	Description
Boolean	removeGovernor	Stores confirmation on the removal in progress to prevent it continuing if there's an error or if the user cancels. Similar to myLocalError.
String	myLocalString	Used to store the MySQL string query locally before set in the database class.
String	myGov	Stores the governor selected from the combo box.
String	governor	Used to hold a concatenated governor record containing governor name, governor ID.
String	myGovArray[]	Used to hold myGov after it has been split to get the governor ID.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	i	Used for iteration in for loops.
Integer	numRecords	Stores the amount of records found in a table in the database.

User Confirmation Box

Type	Name	Description
Boolean	removeConfirmation	Used to store a response from the user to ensure that they confirm or deny that they want to continue their action.

Validation

There are no variables worth noting on a global scale in this class.

View Categories

Type	Name	Description
String	catID	Stores the category ID from the database.
String	catName	Stores the category name from the database.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.

View Completed Courses by Course

Type	Name	Description
String	myLocalString	Used to store a response from the user to ensure that they confirm or deny that they want to continue their action.
String	govID	Used to store the governor ID from the database.
String	myCourseArray[]	Used to hold myCourse after it has been split to get the course ID.
String	courseName	Stores the course name from the database.
String	govName	Stores the governor name from the database.
String	course	Stores the concatenated course record before adding it to the combo box.
String	myCourse	Stores the selected course from the combo box.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	myCourseID	Stores the course ID from myCourseArray[].
Integer	i	Used for iteration in for loops.
Date	dateTaken	Stores the date from the database.

View Completed Courses by Governor

Type	Name	Description
String	myLocalString	Used to store a response from the user to ensure that they confirm or deny that they want to continue their action.
String	courseID	Used to store the course ID from the database.
String	myGovernorArray[]	Used to hold myGovernor after it has been split to get the course ID.
String	courseName	Stores the course name from the database.
String	governorName	Stores the governor name from the database.
String	governor	Stores the concatenated governor record before adding it to the combo box.
String	myGovernor	Stores the selected governor from the combo box.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.
Integer	myGovernorID	Stores the course ID from myCourseArray[].
Integer	i	Used for iteration in for loops.
Date	dateTaken	Stores the date from the database.

View Courses

Type	Name	Description
String	myLocalString	Used to store a response from the user to ensure that they confirm or deny that they want to continue their action.
String	courseID	Used to store the course ID from the database.
String	catID	Stores the category ID from the database.
String	courseName	Stores the course name form the database.
String	catName	Stores the category name from the database.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.

View Governors

Type	Name	Description
String	govID	Used to store a response from the user to ensure that they confirm or deny that they want to continue their action.
String	firstname	Stores the first name of the governor from the database.
String	surname	Stores the surname of the governor from the database.
Integer	thisError	Used to store the error code retrieved from the database query if something goes wrong on execution of a query.
Integer	numRecords	Stores the amount of records found in a table in the database.
Date	elecDate	Stores the election date retrieved from the database.
Date	endDate	Stores the end date retrieved from the database.

```
package cg41;

import org.jdesktop.application.Application;
import org.jdesktop.application.SingleFrameApplication;

/**
 * The main class of the application.
 */
public class CG41App extends SingleFrameApplication {

    /**
     * At startup create and show the main frame of the application.
     */
    @Override protected void startup() {
        show(new CG41View(this));
    }

    /**
     * This method is to initialize the specified window by injecting resources.
     * Windows shown in our application come fully initialized from the GUI
     * builder, so this additional configuration is not needed.
     */
    @Override protected void configureWindow(java.awt.Window root) {
    }

    /**
     * A convenient static getter for the application instance.
     * @return the instance of CG41App
     */
    public static CG41App getApplication() {
        return Application.getInstance(CG41App.class);
    }

    /**
     * Main method launching the application.
     */

    public static MyDBObject dbObject;
    public static MyDBObject dbObject2;
    public static Validation validateObject;

    public static void main(String[] args) {
        //create the database and validation objects
        dbObject = new MyDBObject();
        dbObject2 = new MyDBObject();
        validateObject = new Validation();
        if( CG41App.dbObject.ErrorCode == 0){
            launch(CG41App.class, args);
        }
    }
}
```

```

package cg41;

import org.jdesktop.application.Action;
import org.jdesktop.application.ResourceMap;
import org.jdesktop.application.SingleFrameApplication;
import org.jdesktop.application.FrameView;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.Timer;
import javax.swing.Icon;
import javax.swing.JDialog;
import javax.swing.JFrame;

/**
 * The application's main frame.
 */
public class CG41View extends FrameView {

    int i;

    public CG41View(SingleFrameApplication app) {
        super(app);

        initComponents();

        // status bar initialization - message timeout, idle icon and busy animation, etc
        ResourceMap resourceMap = getResourceMap();
        int messageTimeout = resourceMap.getInteger("StatusBar.messageTimeout");
        messageTimer = new Timer(messageTimeout, new ActionListener() {

            public void actionPerformed(ActionEvent e) {
            }

        });
        messageTimer.setRepeats(false);
        int busyAnimationRate = resourceMap.getInteger("StatusBar.busyAnimationRate");
        busyIconTimer = new Timer(busyAnimationRate, new ActionListener() {

            public void actionPerformed(ActionEvent e) {
            }

        });
        idleIcon = resourceMap.getIcon("StatusBar.idleIcon");

    }

    @Action
    public void showAboutBox() {
        if (aboutBox == null) {
            JFrame mainFrame = CG41App.getApplication().getMainFrame();
            aboutBox = new CG41AboutBox(mainFrame);
            aboutBox.setLocationRelativeTo(mainFrame);
        }
        CG41App.getApplication().show(aboutBox);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.

```

```

*/
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {

    mainPanel = new javax.swing.JPanel();
    mainTabs = new javax.swing.JTabbedPane();
    govPanel = new javax.swing.JPanel();
    addGovBtn = new javax.swing.JButton();
    remGovBtn = new javax.swing.JButton();
    jButton1 = new javax.swing.JButton();
    remGovBtn1 = new javax.swing.JButton();
    jPanel1 = new javax.swing.JPanel();
    addCompletedCourseBtn = new javax.swing.JButton();
    removeCompletedCourseBtn = new javax.swing.JButton();
    viewCompCourseByGov = new javax.swing.JButton();
    viewCompCourseByCourse = new javax.swing.JButton();
    corCatPanel = new javax.swing.JPanel();
    addCouseBtn = new javax.swing.JButton();
    removeCourseBtn = new javax.swing.JButton();
    removeCatBtn = new javax.swing.JButton();
    addCatBtn = new javax.swing.JButton();
    viewCoursesBtn = new javax.swing.JButton();
    viewCatBtn = new javax.swing.JButton();
    editCatBtn = new javax.swing.JButton();
    editCourseBtn = new javax.swing.JButton();
    coursesLbl = new javax.swing.JLabel();
    categoriesLbl = new javax.swing.JLabel();
    reportPanel = new javax.swing.JPanel();
    exitBtn = new javax.swing.JButton();
    menuBar = new javax.swing.JMenuBar();
    javax.swing.JMenu fileMenu = new javax.swing.JMenu();
    javax.swing.JMenuItem exitMenuItem = new javax.swing.JMenuItem();
    javax.swing.JMenu helpMenu = new javax.swing.JMenu();
    javax.swing.JMenuItem aboutMenuItem = new javax.swing.JMenuItem();

    mainPanel.setName("mainPanel"); // NOI18N

    mainTabs.setName("mainTabs"); // NOI18N

    govPanel.setName("govPanel"); // NOI18N

    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(CG41View.
    class);
    addGovBtn.setText(resourceMap.getString("addGovBtn.text")); // NOI18N
    addGovBtn.setName("addGovBtn"); // NOI18N
    addGovBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            addGovBtnActionPerformed(evt);
        }
    });

    remGovBtn.setText(resourceMap.getString("remGovBtn.text")); // NOI18N
    remGovBtn.setName("remGovBtn"); // NOI18N
    remGovBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {

```

```

        remGovBtnActionPerformed (evt) ;
    }
});

jButton1.setText(resourceMap.getString("jButton1.text")); // NOI18N
jButton1.setName("jButton1"); // NOI18N
jButton1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt) ;
    }
});

remGovBtn1.setText(resourceMap.getString("remGovBtn1.text")); // NOI18N
remGovBtn1.setName("remGovBtn1"); // NOI18N
remGovBtn1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        remGovBtn1ActionPerformed(evt) ;
    }
});

javax.swing.GroupLayout govPanelLayout = new javax.swing.GroupLayout(govPanel);
govPanel.setLayout(govPanelLayout);
govPanelLayout.setHorizontalGroup(
    govPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(govPanelLayout.createSequentialGroup()
            .addGap(185, Short.MAX_VALUE)
            .addComponent(addGovBtn)
            .addComponent(remGovBtn1)
            .addComponent(remGovBtn)
            .addComponent(jButton1))
        .addGap(24, Short.MAX_VALUE))
);
govPanelLayout.setVerticalGroup(
    govPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(govPanelLayout.createSequentialGroup()
            .addGap(185, Short.MAX_VALUE)
            .addComponent(addGovBtn)
            .addComponent(remGovBtn1)
            .addComponent(remGovBtn)
            .addComponent(jButton1))
        .addGap(24, Short.MAX_VALUE))
);

mainTabs.addTab(resourceMap.getString("govPanel.TabConstraints.tabTitle"), govPanel);
// NOI18N

jPanel1.setName("jPanel1"); // NOI18N

addCompletedCourseBtn.setText(resourceMap.getString("addCompletedCourseBtn.text"));
// NOI18N
addCompletedCourseBtn.setName("addCompletedCourseBtn"); // NOI18N
addCompletedCourseBtn.addActionListener(new java.awt.event.ActionListener() {

```



```

        public void actionPerformed(java.awt.event.ActionEvent evt) {
            addCompletedCourseBtnActionPerformed(evt);
        }
    });

    removeCompletedCourseBtn.setText(resourceMap.getString(
        "removeCompletedCourseBtn.text")); // NOI18N
    removeCompletedCourseBtn.setName("removeCompletedCourseBtn"); // NOI18N
    removeCompletedCourseBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            removeCompletedCourseBtnActionPerformed(evt);
        }
    });

    viewCompCourseByGov.setText(resourceMap.getString("viewCompCourseByGov.text")); //
    NOI18N
    viewCompCourseByGov.setName("viewCompCourseByGov"); // NOI18N
    viewCompCourseByGov.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            viewCompCourseByGovActionPerformed(evt);
        }
    });

    viewCompCourseByCourse.setText(resourceMap.getString("viewCompCourseByCourse.text"));
    // NOI18N
    viewCompCourseByCourse.setName("viewCompCourseByCourse"); // NOI18N
    viewCompCourseByCourse.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            viewCompCourseByCourseActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(removeCompletedCourseBtn, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(addCompletedCourseBtn, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(viewCompCourseByCourse)
                    .addComponent(viewCompCourseByGov)
                )
                .addGap(101, 101, 101)
            )
    );
    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(addCompletedCourseBtn)
            )
    );

```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addComponent(removeCompletedCourseBtn)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addComponent(viewCompCourseByCourse)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addComponent(viewCompCourseByGov)
.addContainerGap(24, Short.MAX_VALUE))
);

mainTabs.addTab(resourceMap.getString("jPanel1.TabConstraints.tabTitle"), jPanel1);
// NOI18N

corCatPanel.setName("corCatPanel"); // NOI18N

addCouseBtn.setText(resourceMap.getString("addCouseBtn.text")); // NOI18N
addCouseBtn.setName("addCouseBtn"); // NOI18N
addCouseBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        addCouseBtnActionPerformed(evt);
    }
});

removeCourseBtn.setText(resourceMap.getString("removeCourseBtn.text")); // NOI18N
removeCourseBtn.setName("removeCourseBtn"); // NOI18N
removeCourseBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        removeCourseBtnActionPerformed(evt);
    }
});

removeCatBtn.setText(resourceMap.getString("removeCatBtn.text")); // NOI18N
removeCatBtn.setName("removeCatBtn"); // NOI18N
removeCatBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        removeCatBtnActionPerformed(evt);
    }
});

addCatBtn.setText(resourceMap.getString("addCatBtn.text")); // NOI18N
addCatBtn.setName("addCatBtn"); // NOI18N
addCatBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        addCatBtnActionPerformed(evt);
    }
});

viewCoursesBtn.setText(resourceMap.getString("viewCoursesBtn.text")); // NOI18N
viewCoursesBtn.setName("viewCoursesBtn"); // NOI18N
viewCoursesBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        viewCoursesBtnActionPerformed(evt);
    }
});

viewCatBtn.setText(resourceMap.getString("viewCatBtn.text")); // NOI18N
viewCatBtn.setName("viewCatBtn"); // NOI18N
viewCatBtn.addActionListener(new java.awt.event.ActionListener() {
```

-6-

```

        Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
        117, Short.MAX_VALUE)
        .addComponent(editCourseBtn, javax.swing.GroupLayout.
        Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
        117, Short.MAX_VALUE))
        .addGap(58, 58, 58))
    .addGroup(corCatPanelLayout.createSequentialGroup()
        .addComponent(coursesLbl)
        .addGap(132, 132, 132)))
    .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.
    GroupLayout.Alignment.LEADING)
        .addComponent(categoriesLbl)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
        corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.LEADING)
            .addComponent(editCatBtn, javax.swing.GroupLayout.
            DEFAULT_SIZE, 129, Short.MAX_VALUE)
            .addComponent(addCatBtn, javax.swing.GroupLayout.DEFAULT_SIZE
            , 129, Short.MAX_VALUE))))))
    .addContainerGap())
);

corCatPanelLayout.setVerticalGroup(
    corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, corCatPanelLayout.
    createSequentialGroup()
        .addGap(23, 23, 23)
        .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.BASELINE)
            .addComponent(coursesLbl)
            .addComponent(categoriesLbl))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.BASELINE)
            .addComponent(addCatBtn)
            .addComponent(addCouseBtn))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.BASELINE)
            .addComponent(editCatBtn)
            .addComponent(editCourseBtn))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.LEADING)
            .addComponent(removeCourseBtn)
            .addComponent(removeCatBtn))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addGroup(corCatPanelLayout.createParallelGroup(javax.swing.GroupLayout.
        Alignment.LEADING)
            .addComponent(viewCatBtn, javax.swing.GroupLayout.DEFAULT_SIZE, javax.
            swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addComponent(viewCoursesBtn))
        .addContainerGap())
    );

mainTabs.addTab(resourceMap.getString("corCatPanel.TabConstraints.tabTitle"),
corCatPanel); // NOI18N

```

```
reportPanel.setName("reportPanel"); // NOI18N

javax.swing.GroupLayout reportPanelLayout = new javax.swing.GroupLayout(reportPanel);
reportPanel.setLayout(reportPanelLayout);
reportPanelLayout.setHorizontalGroup(
    reportPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGap(0, 324, Short.MAX_VALUE)
);
reportPanelLayout.setVerticalGroup(
    reportPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGap(0, 160, Short.MAX_VALUE)
);

mainTabs.addTab(resourceMap.getString("reportPanel.TabConstraints.tabTitle"),
reportPanel); // NOI18N

javax.swing.ActionMap actionMap = org.jdesktop.application.Application.getInstance(
cg41.CG41App.class).getContext().getActionMap(CG41View.class, this);
exitBtn.setAction(actionMap.get("quit")); // NOI18N
exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
exitBtn.setName("exitBtn"); // NOI18N

javax.swing.GroupLayout mainPanelLayout = new javax.swing.GroupLayout(mainPanel);
mainPanel.setLayout(mainPanelLayout);
mainPanelLayout.setHorizontalGroup(
    mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addComponent(mainTabs, javax.swing.GroupLayout.DEFAULT_SIZE, 329, Short.
MAX_VALUE)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, mainPanelLayout.
createSequentialGroup()
            .addContainerGap(268, Short.MAX_VALUE)
            .addComponent(exitBtn)
            .addContainerGap())
);
mainPanelLayout.setVerticalGroup(
    mainPanelLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(mainPanelLayout.createSequentialGroup()
            .addComponent(mainTabs, javax.swing.GroupLayout.PREFERRED_SIZE, 188, javax.
swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(exitBtn)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
);

menuBar.setName("menuBar"); // NOI18N

fileMenu.setText(resourceMap.getString("fileMenu.text")); // NOI18N
fileMenu.setName("fileMenu"); // NOI18N

exitMenuItem.setAction(actionMap.get("quit")); // NOI18N
exitMenuItem.setName("exitMenuItem"); // NOI18N
fileMenu.add(exitMenuItem);

menuBar.add(fileMenu);

helpMenu.setText(resourceMap.getString("helpMenu.text")); // NOI18N
helpMenu.setName("helpMenu"); // NOI18N
```

```

        aboutMenuItem.setAction(actionMap.get("showAboutBox")); // NOI18N
        aboutMenuItem.setName("aboutMenuItem"); // NOI18N
        helpMenu.add(aboutMenuItem);

        menuBar.add(helpMenu);

        setComponent(mainPanel);
        setMenuBar(menuBar);
    } // </editor-fold> // GEN-END: initComponents

    private void addGovBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_addGovBtnActionPerformed
        AddGovernor addGovWindow = new AddGovernor();
        addGovWindow.setVisible(true);
        // GEN-LAST:event_addGovBtnActionPerformed

    private void addCouseBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_addCouseBtnActionPerformed
        AddCourse addCourseWindow = new AddCourse();
        addCourseWindow.setVisible(true);
        // GEN-LAST:event_addCouseBtnActionPerformed

    private void addCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_addCatBtnActionPerformed
        AddCategory addCategoryWindow = new AddCategory();
        addCategoryWindow.setVisible(true);
        // GEN-LAST:event_addCatBtnActionPerformed

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_jButton1ActionPerformed
        ViewGovernors viewGovsWindow = new ViewGovernors();
        viewGovsWindow.setVisible(true);
        // GEN-LAST:event_jButton1ActionPerformed

    private void addCompletedCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_addCompletedCourseBtnActionPerformed
        AddCompletedCourse addCompCourseWindow = new AddCompletedCourse();
        addCompCourseWindow.setVisible(true);
        // GEN-LAST:event_addCompletedCourseBtnActionPerformed

    private void viewCoursesBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_viewCoursesBtnActionPerformed
        ViewCourses viewCourseWindow = new ViewCourses();
        viewCourseWindow.setVisible(true);
        // GEN-LAST:event_viewCoursesBtnActionPerformed

    private void viewCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_viewCatBtnActionPerformed
        ViewCategories viewCategoryWindow = new ViewCategories();
        viewCategoryWindow.setVisible(true);
        // GEN-LAST:event_viewCatBtnActionPerformed

    private void remGovBtn1ActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_remGovBtn1ActionPerformed
        EditGovernor editGovernorWindow = new EditGovernor();
        editGovernorWindow.setVisible(true);

```

```
//GEN-LAST:event_remGovBtnActionPerformed

private void removeCompletedCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_removeCompletedCourseBtnActionPerformed
    RemoveCompletedCourse removeCompCourseWindow = new RemoveCompletedCourse();
    removeCompCourseWindow.setVisible(true);
}//GEN-LAST:event_removeCompletedCourseBtnActionPerformed

private void viewCompCourseByGovActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_viewCompCourseByGovActionPerformed
    ViewCompletedCoursesByGovernor viewCompCourseGovWindow = new
    ViewCompletedCoursesByGovernor();
    viewCompCourseGovWindow.setVisible(true);
}//GEN-LAST:event_viewCompCourseByGovActionPerformed

private void viewCompCourseByCourseActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_viewCompCourseByCourseActionPerformed
    ViewCompletedCoursesByCourse viewCompCourseCourseWindow = new
    ViewCompletedCoursesByCourse();
    viewCompCourseCourseWindow.setVisible(true);
}//GEN-LAST:event_viewCompCourseByCourseActionPerformed

private void removeCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_removeCatBtnActionPerformed
    RemoveCategory removeCategoryWindow = new RemoveCategory();
    removeCategoryWindow.setVisible(true);
}//GEN-LAST:event_removeCatBtnActionPerformed

private void removeCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_removeCourseBtnActionPerformed
    RemoveCourse removeCourseWindow = new RemoveCourse();
    removeCourseWindow.setVisible(true);
}//GEN-LAST:event_removeCourseBtnActionPerformed

private void remGovBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_remGovBtnActionPerformed
    RemoveGovernor removeGovernorWindow = new RemoveGovernor();
    removeGovernorWindow.setVisible(true);
}//GEN-LAST:event_remGovBtnActionPerformed

private void editCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_editCatBtnActionPerformed
    EditCategory editCategoryWindow = new EditCategory();
    editCategoryWindow.setVisible(true);
}//GEN-LAST:event_editCatBtnActionPerformed

private void editCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_editCourseBtnActionPerformed
    EditCourse editCourseWindow = new EditCourse();
    editCourseWindow.setVisible(true);
}//GEN-LAST:event_editCourseBtnActionPerformed
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton addCatBtn;
private javax.swing.JButton addCompletedCourseBtn;
private javax.swing.JButton addCourseBtn;
private javax.swing.JButton addGovBtn;
private javax.swing.JLabel categoriesLbl;
```



```
private javax.swing.JPanel corCatPanel;  
private javax.swing.JLabel coursesLbl;  
private javax.swing.JButton editCatBtn;  
private javax.swing.JButton editCourseBtn;  
private javax.swing.JButton exitBtn;  
private javax.swing.JPanel govPanel;  
private javax.swing.JButton jButton1;  
private javax.swing.JPanel jPanel1;  
private javax.swing.JPanel mainPanel;  
private javax.swing.JTabbedPane mainTabs;  
private javax.swing.JMenuBar menuBar;  
private javax.swing.JButton remGovBtn;  
private javax.swing.JButton remGovBtn1;  
private javax.swing.JButton removeCatBtn;  
private javax.swing.JButton removeCompletedCourseBtn;  
private javax.swing.JButton removeCourseBtn;  
private javax.swing.JPanel reportPanel;  
private javax.swing.JButton viewCatBtn;  
private javax.swing.JButton viewCompCourseByCourse;  
private javax.swing.JButton viewCompCourseByGov;  
private javax.swing.JButton viewCoursesBtn;  
// End of variables declaration//GEN-END:variables  
private final Timer messageTimer;  
private final Timer busyIconTimer;  
private final Icon idleIcon;  
private final Icon[] busyIcons = new Icon[15];  
private int busyIconIndex = 0;  
private JDialog aboutBox;
```

```
}
```



```
package cg41;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

/**
 * This class was given to me by my teacher.
 * @author cselby
 *
 * This object is my custom database object. I will use it to access the
 * MySQL server where the database resides.
 */
public class MyDBObject {

    private Connection conn = null;
    public Statement stmt = null;
    public int ErrorCode = 0;           // Default place for caller to look for error
    public String sqlString;           // For executing request
    public ResultSet rs = null;        // For returning records
    public int NumberOfRecords = 0;    // For returning number of records found
    public int LastInsertedKey = 0;    // For insert requests only, the key generated

    public int deleteRecord() {
        ErrorCode = 0;                 // No error so far
        try {
            if (rs != null) {
                rs.close();            // Clean up before starting again
                rs = null;
            }
            if (stmt != null) {
                stmt.close();          // Clean up
                stmt = null;
            }
            stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.
CONCUR_UPDATABLE);
            stmt.execute(sqlString);
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            ErrorCode = ex.getErrorCode();

            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
        return (ErrorCode);
    }

    public int updateRecord() {
        ErrorCode = 0;                 // No error so far
        try {
            if (rs != null) {
```

```
        rs.close(); // Clean up before starting again
        rs = null;
    }
    if (stmt != null) {
        stmt.close(); // Clean up
        stmt = null;
    }
    stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.
CONCUR_UPDATABLE);
    stmt.execute(sqlString);
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    ErrorCode = ex.getErrorCode();

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
return (ErrorCode);
}

public int insertRecord() {
    // You can check LastInsertedKey after this call to find the
    // autonumber field that was added to the table
    ErrorCode = 0; // No error so far
    try {
        if (rs != null) {
            rs.close(); // Clean up before starting again
            rs = null;
        }
        if (stmt != null) {
            stmt.close(); // Clean up
            stmt = null;
        }
        stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.
CONCUR_UPDATABLE);
        stmt.execute(sqlString, Statement.RETURN_GENERATED_KEYS);
        rs = stmt.getGeneratedKeys();
        if (rs.next()) {
            LastInsertedKey = rs.getInt(1);
        }
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        ErrorCode = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
    return (ErrorCode);
}
```

```

public int getRecordSetBySelect () {
    ErrorCode = 0; // No error so far
    try {
        if (rs != null) {
            rs.close(); // Clean up before starting again
            rs = null;
        }
        if (stmt != null) {
            stmt.close(); // Clean up
            stmt = null;
        }
        stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.
            CONCUR_UPDATABLE);
        rs = stmt.executeQuery(sqlString);
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        ErrorCode = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
    return (ErrorCode);
}

```

```

public int getCountBySelect () {
    /* This function sets the RecordCount variable in the object in
    * preparation for a later call. Use this when you want to know
    * how many objects your next call is going to return.
    * This is useful for preparing arrays to receive the data
    * */
    ErrorCode = 0; // No error so far
    try {
        if (rs != null) {
            rs.close(); // Clean up before starting again
            rs = null;
        }
        if (stmt != null) {
            stmt.close(); // Clean up
            stmt = null;
        }
        stmt = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.
            CONCUR_UPDATABLE);
        rs = stmt.executeQuery(sqlString);
        rs.first(); // Point to first record
        NumberOfRecords = rs.getInt(1); // Number of records possible
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        ErrorCode = ex.getErrorCode();
    }
}

```

```
        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
    return (ErrorCode);
}
// My customised constructor, which will make a database connection

public MyDBObject () {
    ErrorCode = 0; // Start with no error state
    try {
        conn = DriverManager.getConnection(
            // This is for the local server
            "jdbc:mysql://localhost/cg4", "root", "password");

        // This is for the server at school
        "jdbc:mysql://192.168.0.53:3306/Robinson_John",
        "Robinson_John", "209160");
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        ErrorCode = ex.getErrorCode(); // For whomever called us
        if (ErrorCode == 0) {
            ErrorCode = 1;
        }
        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
}
}
```

```
package cg41;

/**
 * This class was given to me by my teacher.
 * @author cselby
 */
public class UserMessageBox extends javax.swing.JFrame {

    /** Creates new form UserMessageBox */
    public UserMessageBox() {
        initComponents();

        txtMessageOutput.setBorder(null);
    }

    public void setMessage(String Message) {
        txtMessageOutput.setText(Message);
    }

    /** This method is called from within the constructor to
     * initialise the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        okBtn = new javax.swing.JButton();
        txtMessageOutput = new javax.swing.JTextArea();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setName("Form"); // NOI18N

        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
        UserMessageBox.class);
        okBtn.setText(resourceMap.getString("okBtn.text")); // NOI18N
        okBtn.setName("okBtn"); // NOI18N
        okBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                okBtnActionPerformed(evt);
            }
        });

        txtMessageOutput.setBackground(resourceMap.getColor("txtMessageOutput.background"));
        // NOI18N
        txtMessageOutput.setColumns(20);
        txtMessageOutput.setEditable(false);
        txtMessageOutput.setLineWrap(true);
        txtMessageOutput.setRows(5);
        txtMessageOutput.setBorder(null);
        txtMessageOutput.setName("txtMessageOutput"); // NOI18N

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
```

```

        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap(390, Short.MAX_VALUE)
            .addComponent(okBtn)
            .addContainerGap())
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(txtMessageOutput, javax.swing.GroupLayout.PREFERRED_SIZE, 427,
                javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.
            createSequentialGroup()
                .addContainerGap()
                .addComponent(txtMessageOutput, javax.swing.GroupLayout.DEFAULT_SIZE, 223,
                    Short.MAX_VALUE)
                .addGap(18, 18, 18)
                .addComponent(okBtn)
                .addContainerGap())
    );

    pack();
} // </editor-fold> // GEN-END: initComponents

private void okBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST:event_okBtnActionPerformed
    dispose();
    // GEN-LAST:event_okBtnActionPerformed

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {

            public void run() {
                new UserMessageBox().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify // GEN-BEGIN:variables
    private javax.swing.JButton okBtn;
    private javax.swing.JTextArea txtMessageOutput;
    // End of variables declaration // GEN-END:variables
}

```

```

package cg41;

public class UserConfirmationBox extends javax.swing.JDialog {

    public boolean removeConfirmation;

    /** Creates new form UserConfirmationBox */
    public UserConfirmationBox(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
    }

    public void setMessage(String Message) {
        txtMessageOutput.setText(Message);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code"> //GEN-BEGIN: initComponents
    private void initComponents() {

        yesBtn = new javax.swing.JButton();
        noBtn = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        txtMessageOutput = new javax.swing.JTextArea();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setModal(true);
        setModalityType(java.awt.Dialog.ModalityType.DOCUMENT_MODAL);
        setName("Form"); // NOI18N

        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
        UserConfirmationBox.class);
        yesBtn.setText(resourceMap.getString("yesBtn.text")); // NOI18N
        yesBtn.setName("yesBtn"); // NOI18N
        yesBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                yesBtnActionPerformed(evt);
            }
        });

        noBtn.setText(resourceMap.getString("noBtn.text")); // NOI18N
        noBtn.setName("noBtn"); // NOI18N
        noBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                noBtnActionPerformed(evt);
            }
        });

        jScrollPane1.setName("jScrollPane1"); // NOI18N

        txtMessageOutput.setBackground(resourceMap.getColor("txtMessageOutput.background"));
    }
}

```

```

// NOI18N
txtMessageOutput.setColumns(20);
txtMessageOutput.setEditable(false);
txtMessageOutput.setLineWrap(true);
txtMessageOutput.setRows(5);
txtMessageOutput.setName("txtMessageOutput"); // NOI18N
jScrollPane1.setViewportView(txtMessageOutput);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.
            createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
                    TRAILING)
                    .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 357,
                        Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(yesBtn)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(noBtn)))
                .addContainerGap()
        );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
                TRAILING, false)
                .addComponent(yesBtn, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(noBtn, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addContainerGap()
        );

pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * When the yes button is pressed, this method will set the public
 * remove confirmation variable to true. It will also hide this window and
 * allow the you to retrieve the variable before closing this object
 */
private void yesBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_yesBtnActionPerformed
    removeConfirmation = true;
    setVisible(false);
} // GEN-LAST: event_yesBtnActionPerformed

```



```

/**
 * When the no button is pressed, this method will set the public
 * remove confirmation variable to false. It will also hide this window and
 * allow the you to retrieve the variable before closing this object
 */
private void noBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_noBtnActionPerformed
    removeConfirmation = false;
    setVisible(false);
}//GEN-LAST:event_noBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            UserConfirmationBox dialog = new UserConfirmationBox(new javax.swing.JFrame
            (), true);
            dialog.addWindowListener(new java.awt.event.WindowAdapter() {

                public void windowClosing(java.awt.event.WindowEvent e) {
                    System.exit(0);
                }
            });
            dialog.setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JButton noBtn;
private javax.swing.JTextArea txtMessageOutput;
private javax.swing.JButton yesBtn;
// End of variables declaration//GEN-END:variables
}

```

```
package cg41;

import javax.swing.WindowConstants;

public class AddCategory extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString, errorMessage = null;
    int thisError = 0;

    /** Creates new form AddGovernor */
    public AddCategory() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialise the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        catIdOutLbl = new javax.swing.JLabel();
        clrFldsBtn = new javax.swing.JButton();
        catNameLbl = new javax.swing.JLabel();
        addCatBtn = new javax.swing.JButton();
        catIdLbl = new javax.swing.JLabel();
        catNameFld = new javax.swing.JTextField();
        exitBtn = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(AddCategory.
        class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        catIdOutLbl.setText(resourceMap.getString("catIdOutLbl.text")); // NOI18N
        catIdOutLbl.setName("catIdOutLbl"); // NOI18N

        clrFldsBtn.setText(resourceMap.getString("clrFldsBtn.text")); // NOI18N
        clrFldsBtn.setName("clrFldsBtn"); // NOI18N
        clrFldsBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                clrFldsBtnActionPerformed(evt);
            }
        });

        catNameLbl.setText(resourceMap.getString("catNameLbl.text")); // NOI18N
        catNameLbl.setName("catNameLbl"); // NOI18N
```

-2-

```

        .TRAILING)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.
            Alignment.BASELINE)
            .addComponent(catNameLbl)
            .addComponent(catNameFld, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                    PREFERRED_SIZE))
        .addGroup(jPanel1Layout.createSequentialGroup())
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.
            Alignment.BASELINE)
            .addComponent(catIdLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
                17, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(catIdOutLbl))
        .addGap(5, 5, 5)))
    .addGap(48, 48, 48)
    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
        .BASELINE)
        .addComponent(exitBtn)
        .addComponent(addCatBtn)
        .addComponent(clrFldsBtn))
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
);

```

```

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
    );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

```

```

jPanel1.getAccessibleContext().setAccessibleName(resourceMap.getString(
    "jPanel1.AccessibleContext.accessibleName")); // NOI18N

```

```
pack();
```

```
}// </editor-fold>//GEN-END:initComponents
```

```

private void addCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_addCatBtnActionPerformed
    myLocalError = false; //set variables used to null
    thisError = 0;
    myLocalString = "";
    if ((catNameFld.getText().equals("")))) { //check for blank field
        myLocalError = true;
    }
    if (myLocalError) {

```

```

    // field must be blank, so error
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Please Enter a Name for the Category.");
    MyMessage.setTitle("Error Adding Category");
    MyMessage.setVisible(true);
}

errorMessage = "Please check the following: \n\n";
if (!myLocalError) { //if there's no error
    //validate category name
    if (!CG41App.validateObject.validateCategoryName(catNameFld.getText())) {
        myLocalError = true; //set error true and add to the error message
        errorMessage = errorMessage + "Category name should be no longer than 22
            characters and"
            + "must begin with a letter, space, dash or apostrophe.";
    }
    if (myLocalError) {
        //One of the validation checks failed
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage(errorMessage);
        MyMessage.setTitle("Error Adding Category");
        MyMessage.setVisible(true);
    }
}

if (!myLocalError) { //add in the database
    myLocalString = "insert into category (" + "CategoryName)" + "values (";
    myLocalString = myLocalString + "'" + catNameFld.getText() + "'" + ") ";
    CG41App.dbObject.sqlString = myLocalString; //set the SQL string
    thisError = CG41App.dbObject.insertRecord(); //execute string

    if (0 == thisError) {
        //adding was successful
        catIdOutLbl.setText(Integer.toString(CG41App.dbObject.LastInsertedKey));
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage("Category Added Edited.");
        MyMessage.setTitle("Category Edited");
    }
}

}

} //GEN-LAST:event_addCatBtnActionPerformed
/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} //GEN-LAST:event_exitBtnActionPerformed
/**
 * this method clears the category field for more input
 */
private void clrFldsBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_clrFldsBtnActionPerformed
    catNameFld.setText("");
} //GEN-LAST:event_clrFldsBtnActionPerformed

```

```

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(AddCategory.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(AddCategory.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(AddCategory.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(AddCategory.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new AddCategory().setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton addCatBtn;
private javax.swing.JLabel catIdLbl;
private javax.swing.JLabel catIdOutLbl;
private javax.swing.JTextField catNameFld;
private javax.swing.JLabel catNameLbl;
private javax.swing.JButton clrFldsBtn;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
// End of variables declaration//GEN-END:variables
}

```

```
package cg41;

import java.sql.SQLException;
import java.util.Date;
import javax.swing.WindowConstants;

public class AddCompletedCourse extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString = "";
    String governor, courseCompDateString, course;
    String compGov, completedCourse;
    String compCourseIDArray[], compGovIDArray[];
    Date courseDate;
    int thisError, numRecords, courseID, govID, i = 0;

    /**
     * This method loads the courses into the combo box
     */
    private void loadCourses() {
        CG41App.dbObject.sqlString = "select COUNT(*) from course"; //set query
        thisError = CG41App.dbObject.getCountBySelect(); //execute query
        numRecords = CG41App.dbObject.NumberOfRecords; //get number of records
        CG41App.dbObject.sqlString = "select CourseID, CourseName from course";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
        if (thisError == 0) {
            try {
                for (i = 0; i < numRecords; i++) {
                    //for amount of records, add each one to the combo box
                    CG41App.dbObject.rs.next();
                    course = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2);
                    compCourseCombox.addItem(course);
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /**
     * This method loads the governors into the combo box
     */
    private void loadGovernors() {
        CG41App.dbObject.sqlString = "select COUNT(*) from governor"; //get count
        thisError = CG41App.dbObject.getCountBySelect(); //execute count query
        numRecords = CG41App.dbObject.NumberOfRecords; //get number of records
        CG41App.dbObject.sqlString = "select GovID, Firstname, Surname from governor";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
```

```

        if (thisError == 0) {
            try {
                for (i = 0; i < numRecords; i++) {
                    //concatenate governor and add to the combo box for amount of records
                    CG41App.dbObject.rs.next();
                    governor = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2) + " "
                        + CG41App.dbObject.rs.getString(3);
                    compGovCombox.addItem(governor);
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

/**
 * Constructor for AddCompletedCourse class
 */
public AddCompletedCourse() {
    initComponents();
    loadCourses();
    loadGovernors();
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    clrFldsBtn = new javax.swing.JButton();
    addCompletedCourseBtn = new javax.swing.JButton();
    exitBtn = new javax.swing.JButton();
    courseCompletedIDLbl = new javax.swing.JLabel();
    courseCompletedIDOutLbl = new javax.swing.JLabel();
    compGovLbl = new javax.swing.JLabel();
    compGovCombox = new javax.swing.JComboBox();
    compCourseCombox = new javax.swing.JComboBox();
    compCourseLbl = new javax.swing.JLabel();
    compDateLbl = new javax.swing.JLabel();
    compCourseDateChooser = new com.toedter.calendar.JDateChooser();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.

```



```
Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
AddCompletedCourse.class);
setTitle(resourceMap.getString("Form.title")); // NOI18N
setName("Form"); // NOI18N

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
"jPanel1.border.title"))); // NOI18N
jPanel1.setName("jPanel1"); // NOI18N

clrFldsBtn.setText(resourceMap.getString("clrFldsBtn.text")); // NOI18N
clrFldsBtn.setName("clrFldsBtn"); // NOI18N

addCompletedCourseBtn.setText(resourceMap.getString("addCompletedCourseBtn.text"));
// NOI18N
addCompletedCourseBtn.setName("addCompletedCourseBtn"); // NOI18N
addCompletedCourseBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        addCompletedCourseBtnActionPerformed(evt);
    }
});

exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
exitBtn.setName("exitBtn"); // NOI18N
exitBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exitBtnActionPerformed(evt);
    }
});

courseCompletedIDLbl.setText(resourceMap.getString("courseCompletedIDLbl.text")); //
NOI18N
courseCompletedIDLbl.setName("courseCompletedIDLbl"); // NOI18N

courseCompletedIDOutLbl.setText(resourceMap.getString("courseCompletedIDOutLbl.text"
)); // NOI18N
courseCompletedIDOutLbl.setName("courseCompletedIDOutLbl"); // NOI18N

compGovLbl.setText(resourceMap.getString("compGovLbl.text")); // NOI18N
compGovLbl.setName("compGovLbl"); // NOI18N

compGovCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
compGovCombox.setName("compGovCombox"); // NOI18N

compCourseCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
compCourseCombox.setToolTipText(resourceMap.getString("compCourseCombox.toolTipText"
)); // NOI18N
compCourseCombox.setName("compCourseCombox"); // NOI18N

compCourseLbl.setText(resourceMap.getString("compCourseLbl.text")); // NOI18N
compCourseLbl.setName("compCourseLbl"); // NOI18N

compDateLbl.setText(resourceMap.getString("compDateLbl.text")); // NOI18N
compDateLbl.setName("compDateLbl"); // NOI18N

compCourseDateChooser.setName("compCourseDateChooser"); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
```

```

jPanell.setLayout(jPanellLayout);
jPanellLayout.setHorizontalGroup(
    jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanellLayout.createSequentialGroup())
            .addContainerGap()
            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanellLayout.createSequentialGroup())
                    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(compGovLbl)
                        .addComponent(compCourseLbl)
                        .addComponent(compDateLbl))
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(compCourseDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGroup(jPanellLayout.createSequentialGroup())
                            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                                .addComponent(compCourseCombox, javax.swing.GroupLayout.Alignment.LEADING, 0, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(compGovCombox, javax.swing.GroupLayout.Alignment.LEADING, 0, 197, Short.MAX_VALUE))
                            .addGap(28, 28, 28)
                            .addComponent(courseCompletedIDLbl))
                        .addComponent(courseCompletedIDOutLbl, javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.PREFERRED_SIZE, 51, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanellLayout.createSequentialGroup())
                    .addComponent(clrFldsBtn)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(addCompletedCourseBtn, javax.swing.GroupLayout.PREFERRED_SIZE, 89, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(exitBtn)))
            .addContainerGap())
);
jPanellLayout.setVerticalGroup(
    jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanellLayout.createSequentialGroup())
            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanellLayout.createSequentialGroup())
                    .addContainerGap()
                    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .addComponent(compGovCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(compGovLbl))
                    .addGroup(jPanellLayout.createSequentialGroup())

```

```

        .addGap(18, 18, 18)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.
            Alignment.BASELINE)
            .addComponent(compCourseCombox, javax.swing.GroupLayout.
                PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.PREFERRED_SIZE)
            .addComponent(compCourseLbl))
        .addGap(18, 18, 18)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.
            Alignment.TRAILING)
            .addComponent(compCourseDateChooser, javax.swing.GroupLayout.
                PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.PREFERRED_SIZE)
            .addComponent(compDateLbl)))
        .addGroup(jPanel1Layout.createSequentialGroup())
            .addComponent(courseCompletedIDLbl, javax.swing.GroupLayout.
                PREFERRED_SIZE, 17, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(7, 7, 7)
            .addComponent(courseCompletedIDOutLbl)))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 22,
            Short.MAX_VALUE)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
            BASELINE)
            .addComponent(exitBtn)
            .addComponent(addCompletedCourseBtn)
            .addComponent(clrFldsBtn))
        .addContainerGap())
    );

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
    );

    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                    GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap())
    );

    jPanel1.getAccessibleContext().setAccessibleName(resourceMap.getString(
        "jPanel1.AccessibleContext.accessibleName")); // NOI18N

    pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * When the Add Comp Course button is pressed, the fields are checked
 * and then the information is added to the database

```

```

*/
private void addCompletedCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_addCompletedCourseBtnActionPerformed
myLocalError = false; //set used variables to null
thisError = 0;
myLocalString = "";

courseDate = compCourseDateChooser.getDate(); //get date and format
courseCompDateString = String.format("%1$tY-%1$tm-%1$td", courseDate);
//This if statement will check to make sure the user has selected everything
if(courseCompDateString.equals("null-null-null")
    || compGovCombox.getSelectedItem().equals("-")
    || compCourseCombox.getSelectedItem().equals("-")){

myLocalError = true;
//error message for the user
UserMessageBox MyMessage = new UserMessageBox();
MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
MyMessage.setMessage("Error adding completed course.\n"
    + "Please make sure you have selected something\nin all fields.\n"
    + "Please try again.");
MyMessage.setTitle("Error Adding Completed Course");
MyMessage.setVisible(true);

}
else { //if all is okay
compGov = compGovCombox.getSelectedItem().toString(); //get governor
compGovIDArray = compGov.split(",", 2); //split string
govID = Integer.parseInt(compGovIDArray[0]); //get the gov ID
//same as with the gov but with the course
completedCourse = compCourseCombox.getSelectedItem().toString();
compCourseIDArray = completedCourse.split(",", 2);
courseID = Integer.parseInt(compCourseIDArray[0]);

}

if(myLocalError == false){ //add the comp course if there's no error
myLocalString = "insert into coursetaken ("
    + "GovID, CourseID, DateTaken)"
    + "values ('" + govID + "', "
    + "'" + courseID + "', "
    + "'" + courseCompDateString + "');" //concatenate string
CG41App.dbObject.sqlString = myLocalString; //set string
thisError = CG41App.dbObject.insertRecord(); //execute query
}
if (0 == thisError && myLocalError == false) {
//if there's no error, show confirmation box to user
courseCompletedIDOutLbl.setText(Integer.toString(CG41App.dbObject.LastInsertedKey));
UserMessageBox MyMessage = new UserMessageBox();
MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
MyMessage.setMessage("Completed Coruse successfully added.");
MyMessage.setTitle("Completed Course Added");
MyMessage.setVisible(true);
}
else{
//if there's an error then show an error
UserMessageBox MyMessage = new UserMessageBox();
MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
}
}

```

```

        MyMessage.setMessage("There was an error adding the completed course.");
        MyMessage.setTitle("Error Adding Completed Course");
        MyMessage.setVisible(true);
    }
} //GEN-LAST:event_addCompletedCourseBtnActionPerformed
/**
 * This method disposes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} //GEN-LAST:event_exitBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(AddCompletedCourse.class.getName()).log(java.
        util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(AddCompletedCourse.class.getName()).log(java.
        util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(AddCompletedCourse.class.getName()).log(java.
        util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(AddCompletedCourse.class.getName()).log(java.
        util.logging.Level.SEVERE, null, ex);
    }
} //</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {

    public void run() {
        new AddCompletedCourse().setVisible(true);
    }
});

```

```
}  
// Variables declaration - do not modify//GEN-BEGIN:variables  
private javax.swing.JButton addCompletedCourseBtn;  
private javax.swing.JButton clrFldsBtn;  
private javax.swing.JComboBox compCourseCombox;  
private com.toedter.calendar.JDateChooser compCourseDateChooser;  
private javax.swing.JLabel compCourseLbl;  
private javax.swing.JLabel compDateLbl;  
private javax.swing.JComboBox compGovCombox;  
private javax.swing.JLabel compGovLbl;  
private javax.swing.JLabel courseCompletedIDLbl;  
private javax.swing.JLabel courseCompletedIDOutLbl;  
private javax.swing.JButton exitBtn;  
private javax.swing.JPanel jPanel1;  
// End of variables declaration//GEN-END:variables  
}
```

```
package cg41;

import java.sql.SQLException;
import javax.swing.WindowConstants;

public class AddCourse extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString = "";
    String catSelect, errorMessage = "";
    int thisError = 0;
    int numRecords = 0;
    int i = 0;
    int catIDSelect = 0;

    /** Creates new form AddCourse */
    public AddCourse() {
        initComponents();
        loadCatComboBox();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        courseIdOutLbl = new javax.swing.JLabel();
        clrFldsBtn = new javax.swing.JButton();
        catSelectLbl = new javax.swing.JLabel();
        courseNameLbl = new javax.swing.JLabel();
        addCourseBtn = new javax.swing.JButton();
        courseIdLbl = new javax.swing.JLabel();
        courseNameFld = new javax.swing.JTextField();
        exitBtn = new javax.swing.JButton();
        catComBox = new javax.swing.JComboBox();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(AddCourse.
        class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        courseIdOutLbl.setText(resourceMap.getString("courseIdOutLbl.text")); // NOI18N
        courseIdOutLbl.setName("courseIdOutLbl"); // NOI18N

        clrFldsBtn.setText(resourceMap.getString("clrFldsBtn.text")); // NOI18N
        clrFldsBtn.setName("clrFldsBtn"); // NOI18N
```

-2-


```

        .addComponent(courseNameFld, javax.swing.GroupLayout.Alignment.
LEADING, javax.swing.GroupLayout.DEFAULT_SIZE, 137, Short.
MAX_VALUE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(courseIdLbl)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(courseIdOutLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
36, javax.swing.GroupLayout.PREFERRED_SIZE)))
        .addContainerGap())
);

jPanell1Layout.setVerticalGroup(
jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(jPanell1Layout.createSequentialGroup())
.addContainerGap())
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.
Alignment.BASELINE)
.addComponent(courseNameLbl)
.addComponent(courseNameFld, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
PREFERRED_SIZE))
.addGroup(jPanell1Layout.createSequentialGroup())
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.
Alignment.BASELINE)
.addComponent(courseIdLbl, javax.swing.GroupLayout.PREFERRED_SIZE
, 17, javax.swing.GroupLayout.PREFERRED_SIZE)
.addComponent(courseIdOutLbl))
.addGap(5, 5, 5)))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
.addComponent(catSelectLbl)
.addComponent(catComBox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(73, 73, 73)
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
.addComponent(exitBtn)
.addComponent(addCourseBtn)
.addComponent(clrFldsBtn))
.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())
            .addContainerGap()
            .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, 414, Short.
MAX_VALUE)
            .addContainerGap())
);

layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())

```

```

        .addContainerGap ()
        .addComponent (jPanell1, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
        GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
        .addContainerGap (javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

    jPanell1.getAccessibleContext ().setAccessibleName (resourceMap.getString (
    "jPanell1.AccessibleContext.accessibleName")); // NOI18N

    pack ();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method loads the categories from the database into the combo box
 */
private void loadCatComboBox () {
    CG41App.dbObject.sqlString = "select COUNT(*) from category";
    thisError = CG41App.dbObject.getCountBySelect (); // count the categories
    numRecords = CG41App.dbObject.NumberOfRecords; // get the amount of categories
    CG41App.dbObject.sqlString = "select CatID, CategoryName from category";
    thisError = CG41App.dbObject.getRecordSetBySelect (); // execute query
    if (thisError == 0) {
        try {
            for (i = 0; i < numRecords; i++) { // for the amount of records, add each
            category
                CG41App.dbObject.rs.next ();
                // add only the name to the field
                catComBox.addItem (CG41App.dbObject.rs.getString (2));
            }
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println ("SQLException: " + ex.getMessage ());
            System.out.println ("SQLState: " + ex.getSQLState ());
            System.out.println ("VendorError: " + ex.getErrorCode ());
            thisError = ex.getErrorCode ();

            UserMessageBox MyMessage = new UserMessageBox ();
            MyMessage.setMessage ("SQLException: " + ex.getMessage ());
            MyMessage.setVisible (true);
        }
    }
}

/**
 * This method will, when clicked, add the course and its category to the data base
 */
private void addCourseBtnActionPerformed (java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_addCourseBtnActionPerformed
    myLocalError = false; // set used variables to null
    myLocalString = null;
    thisError = 0;
    numRecords = 0;
    catIDSelect = 0;
    catSelect = "";

    if (courseNameFld.getText ().equals ("") // check for blank fields
        || catComBox.getSelectedItem ().toString ().equals ("")) {

```

```

myLocalError = true;
//error for blank fields
UserMessageBox MyMessage = new UserMessageBox();
MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
MyMessage.setMessage("Please make sure you have filled out all the fields");
MyMessage.setTitle("Error Adding Course");
MyMessage.setVisible(true);
}
errorMessage = "Please check the following: \n\n";
if (!myLocalError) {
    //validation of course name
    if (!CG41App.validateObject.validateCourseName(courseNameFld.getText())) {
        myLocalError = true;
        errorMessage = errorMessage + "The course name should begin with"
            + "a letter and be followed by up to 20 letters, numbers"
            + "apostrophes, spaces and dashes.";
    }
    if (myLocalError) {
        //One of the validation checks failed
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage(errorMessage);
        MyMessage.setTitle("Error Adding Course");
        MyMessage.setVisible(true);
    }
}
//create and set SQL string
CG41App.dbObject.sqlString = "select COUNT(*) from category";
thisError = CG41App.dbObject.getCountBySelect();
numRecords = CG41App.dbObject.NumberOfRecords;
CG41App.dbObject.sqlString = "select CatID, CategoryName from category";
thisError = CG41App.dbObject.getRecordSetBySelect(); //execute string

catSelect = catComBox.getSelectedItem().toString(); //get cat ID from combo box
for (i = 0; i < numRecords; i++) {
    try {
        CG41App.dbObject.rs.next(); //find the CatID from the DB search
        if (catSelect.equals(CG41App.dbObject.rs.getString(2))) {
            catIDSelect = CG41App.dbObject.rs.getInt(1);
        }
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        thisError = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
}
if (!myLocalError) { //add course to the DB
    myLocalString = "insert into course (" + "CatID, CourseName)"
        + "values (" + "'" + catIDSelect + "', "
        + "'" + courseNameFld.getText() + "'" + ") "; //concatenate string
    CG41App.dbObject.sqlString = myLocalString; //set string
}

```

```

thisError = CG41App.dbObject.insertRecord(); //execute

if (0 == thisError) {
    //if adding is successful
    courseIdOutLbl.setText(Integer.toString(CG41App.dbObject.LastInsertedKey));
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Course added successfully");
    MyMessage.setTitle("Adding Course Complete");
    MyMessage.setVisible(true);
}
else{
    //if there's an error present
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Error Adding Course.");
    MyMessage.setTitle("Error Adding Course");
    MyMessage.setVisible(true);
}
}
}
//GEN-LAST:event_addCourseBtnActionPerformed
/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_exitBtnActionPerformed
    dispose();
//GEN-LAST:event_exitBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(AddCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(AddCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(AddCourse.class.getName()).log(java.util.

```

```
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(AddCourse.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    }
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {

    public void run() {
        new AddCourse().setVisible(true);
    }
});
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton addCourseBtn;
private javax.swing.JComboBox catComBox;
private javax.swing.JLabel catSelectLbl;
private javax.swing.JButton clrFldsBtn;
private javax.swing.JLabel courseIdLbl;
private javax.swing.JLabel courseIdOutLbl;
private javax.swing.JTextField courseNameFld;
private javax.swing.JLabel courseNameLbl;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
// End of variables declaration//GEN-END:variables
}
```

```
package cg41;

import java.util.Date;
import javax.swing.WindowConstants;

public class AddGovernor extends javax.swing.JFrame {

    boolean myLocalError = false;
    String endDateString, elecDateString, myLocalString, errorMessage = "";
    int thisError = 0;
    Date endDate, elecDate;

    /** Creates new form AddGovernor */
    public AddGovernor() {
        initComponents();
    }

    /** This method is called from within the constructor to
     *  initialise the form.
     *  WARNING: Do NOT modify this code. The content of this method is
     *  always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        govIdOutLbl = new javax.swing.JLabel();
        endDateLbl = new javax.swing.JLabel();
        clrFldsBtn = new javax.swing.JButton();
        endDateChooser = new com.toedter.calendar.JDateChooser();
        lstNamLbl = new javax.swing.JLabel();
        frstNamLbl = new javax.swing.JLabel();
        elecDateChooser = new com.toedter.calendar.JDateChooser();
        addGovBtn = new javax.swing.JButton();
        govIdLbl = new javax.swing.JLabel();
        lstNamFld = new javax.swing.JTextField();
        frstNamFld = new javax.swing.JTextField();
        elecDateLbl = new javax.swing.JLabel();
        exitBtn = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(AddGovernor.
        class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        govIdOutLbl.setText(resourceMap.getString("govIdOutLbl.text")); // NOI18N
        govIdOutLbl.setName("govIdOutLbl"); // NOI18N

        endDateLbl.setText(resourceMap.getString("endDateLbl.text")); // NOI18N
        endDateLbl.setName("endDateLbl"); // NOI18N
```

```
clrFldsBtn.setText(resourceMap.getString("clrFldsBtn.text")); // NOI18N
clrFldsBtn.setName("clrFldsBtn"); // NOI18N
clrFldsBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        clrFldsBtnActionPerformed(evt);
    }
});

endDateChooser.setName("endDateChooser"); // NOI18N

lstNamLbl.setText(resourceMap.getString("lstNamLbl.text")); // NOI18N
lstNamLbl.setName("lstNamLbl"); // NOI18N

frstNamLbl.setText(resourceMap.getString("frstNamLbl.text")); // NOI18N
frstNamLbl.setName("frstNamLbl"); // NOI18N

elecDateChooser.setName("elecDateChooser"); // NOI18N

addGovBtn.setText(resourceMap.getString("addGovBtn.text")); // NOI18N
addGovBtn.setName("addGovBtn"); // NOI18N
addGovBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        addGovBtnActionPerformed(evt);
    }
});

govIdLbl.setText(resourceMap.getString("govIdLbl.text")); // NOI18N
govIdLbl.setName("govIdLbl"); // NOI18N

lstNamFld.setText(resourceMap.getString("lstNamFld.text")); // NOI18N
lstNamFld.setName("lstNamFld"); // NOI18N

frstNamFld.setText(resourceMap.getString("frstNamFld.text")); // NOI18N
frstNamFld.setName("frstNamFld"); // NOI18N

elecDateLbl.setText(resourceMap.getString("elecDateLbl.text")); // NOI18N
elecDateLbl.setName("elecDateLbl"); // NOI18N

exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
exitBtn.setName("exitBtn"); // NOI18N
exitBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exitBtnActionPerformed(evt);
    }
});

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(elecDateLbl)
                .addComponent(frstNamLbl)
            )
        )
);
```

```

        .addComponent(1stNamLbl)
        .addComponent(endDateLbl))
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
    .LEADING)
        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanellLayout.
        createSequentialGroup())
            .addComponent(clrFldsBtn, javax.swing.GroupLayout.DEFAULT_SIZE, javax
            .swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGap(57, 57, 57)
            .addComponent(addGovBtn)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(exitBtn))
    .addComponent(endDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE, 149
    , javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGroup(jPanellLayout.createSequentialGroup())
        .addComponent(frstNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, 137
        , javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(38, 38, 38)
        .addComponent(govIdLbl)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(govIdOutLbl, javax.swing.GroupLayout.DEFAULT_SIZE, 60,
        Short.MAX_VALUE))
    .addComponent(1stNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, 185,
    javax.swing.GroupLayout.PREFERRED_SIZE)
    .addComponent(elecDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
    149, javax.swing.GroupLayout.PREFERRED_SIZE))
    .addContainerGap())
);
jPanellLayout.setVerticalGroup(
    jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanellLayout.createSequentialGroup())
        .addContainerGap()
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
        .TRAILING)
            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.
            Alignment.BASELINE)
                .addComponent(frstNamLbl)
                .addComponent(frstNamFld, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                PREFERRED_SIZE))
            .addGroup(jPanellLayout.createSequentialGroup())
                .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.
                Alignment.BASELINE)
                    .addComponent(govIdLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
                    17, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(govIdOutLbl))
                .addGap(5, 5, 5)))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
        .BASELINE)
            .addComponent(1stNamLbl)
            .addComponent(1stNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, javax
            .swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(18, 18, 18)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
        .LEADING)

```



```

        .addComponent(elecDateLbl)
        .addComponent(elecDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
            javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                PREFERRED_SIZE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .LEADING)
            .addComponent(endDateLbl)
            .addComponent(endDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                    PREFERRED_SIZE))
        .addGap(10, 10, 10)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(exitBtn)
            .addComponent(addGovBtn)
            .addComponent(clrFldsBtn))
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

```

```

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap()
        );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

```

```
pack();
```

```
}// </editor-fold>//GEN-END:initComponents
```

```
/**
```

```
 * This method is used to add a governor on click of the add governor button
```

```
 */
```

```
private void addGovBtnActionPerformed(java.awt.event.ActionEvent evt) {
```

```
//GEN-FIRST:event_addGovBtnActionPerformed
```

```
    myLocalError = false; //set used variables to empty
```

```
    thisError = 0;
```

```
    myLocalString = "";
```

```
    elecDate = elecDateChooser.getDate(); //convert date strings to Dates variables
```

```
    elecDateString = String.format("%1$tY-%1$tm-%1$td", elecDate); //format dates
```

```
    endDate = endDateChooser.getDate();
```

```
    endDateString = String.format("%1$tY-%1$tm-%1$td", endDate);
```

```

//Quick check to see if the fields have been filled in, else print an error
if (firstNamFld.getText().equals(""))
    || lstNamFld.getText().equals("")
    || elecDateString.equals("null-null-null")
    || endDateString.equals("null-null-null")) {
    myLocalError = true;
}
if (myLocalError) {
    // One field must be blank, therefore display an error
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Please Fill Out All Fields.");
    MyMessage.setTitle("Error Adding Governor");
    MyMessage.setVisible(true);
}
//Error checking for validation of each field where an error message will be
//created for the user's viewing for any errors
errorMessage = "Please check the following: \n\n";
if (!myLocalError) {
    //validation of first name
    if(!CG41App.validateObject.validateFirstName(firstNamFld.getText())){
        myLocalError = true;
        errorMessage = errorMessage + "First name should begin with a capital"
            + " and be no longer than 10 characters. \n\n";
    }
    //validation of last name
    if(!CG41App.validateObject.validateSurname(lstNamFld.getText())){
        myLocalError = true;
        errorMessage = errorMessage + "Last name should begin with a letter"
            + " and be no longer than 20 characters and can include"
            + " spaces, dashes and apostrophes. \n\n";
    }
    //check to see if the election date is after the end date
    if (elecDate.after(endDate)) {
        myLocalError = true;
        errorMessage = errorMessage + "Make sure the elected date is before"
            + " the 'end date'.";
    }
    //DB check
    if (!myLocalError) {
        // TODO Check if already in database
    }
    if (myLocalError) {
        //One of the validation checks failed
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage(errorMessage);
        MyMessage.setTitle("Error Adding Governor");
        MyMessage.setVisible(true);
    }
}
if (!myLocalError) { //add the governor to the DB
    myLocalString = "insert into governor (";
    myLocalString = myLocalString + "Firstname, Surname, ElectionDate, EndDate)"
        + " values ("
        + "'" + firstNamFld.getText() + "', "
        + "'" + lstNamFld.getText() + "', "

```

```

        + "" + elecDateString + ", "
        + "" + endDateString + "') ";
CG41App.dbObject.sqlString = myLocalString; //set MySQL string
thisError = CG41App.dbObject.insertRecord(); //execute MySQL string

if (0 == thisError) { //set the new governor ID so the user can see it
    govIdOutLbl.setText(Integer.toString(CG41App.dbObject.LastInsertedKey));
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Governor Added Successfully.");
    MyMessage.setTitle("Governor Added");
    MyMessage.setVisible(true);
}
}
} //GEN-LAST:event_addGovBtnActionPerformed

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} //GEN-LAST:event_exitBtnActionPerformed

/**
 * This method clears all the fields so the user can add another governor
 */
private void clrFldsBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_clrFldsBtnActionPerformed
    frstNamFld.setText("");
    lstNamFld.setText("");
    elecDateChooser.setDate(null);
    endDateChooser.setDate(null);
    govIdOutLbl.setText("-");
} //GEN-LAST:event_clrFldsBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

```

```

        java.util.logging.Logger.getLogger (AddGovernor.class.getName()).log (java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger (AddGovernor.class.getName()).log (java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger (AddGovernor.class.getName()).log (java.util.
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger (AddGovernor.class.getName()).log (java.util.
        logging.Level.SEVERE, null, ex);
    }
}
//</editor-fold>

```

```

/* Create and display the form */

```

```

java.awt.EventQueue.invokeLater(new Runnable() {

```

```

    public void run() {
        new AddGovernor().setVisible(true);
    }
}

```

```

});

```

```

}

```

```

// Variables declaration - do not modify//GEN-BEGIN:variables

```

```

private javax.swing.JButton addGovBtn;

```

```

private javax.swing.JButton clrFldsBtn;

```

```

private com.toedter.calendar.JDateChooser elecDateChooser;

```

```

private javax.swing.JLabel elecDateLbl;

```

```

private com.toedter.calendar.JDateChooser endDateChooser;

```

```

private javax.swing.JLabel endDateLbl;

```

```

private javax.swing.JButton exitBtn;

```

```

private javax.swing.JTextField frstNamFld;

```

```

private javax.swing.JLabel frstNamLbl;

```

```

private javax.swing.JLabel govIdLbl;

```

```

private javax.swing.JLabel govIdOutLbl;

```

```

private javax.swing.JPanel jPanel1;

```

```

private javax.swing.JTextField lstNamFld;

```

```

private javax.swing.JLabel lstNamLbl;

```

```

// End of variables declaration//GEN-END:variables

```

```

}

```

```
package cg41;

import java.sql.SQLException;
import javax.swing.WindowConstants;

public class EditCategory extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString = null;
    String category, myCat, errorMessage = "";
    String myCatArray[];
    int thisError, i, numRecords = 0;

    /** Creates new form AddGovernor */
    public EditCategory() {
        initComponents();
        loadCategories();
    }

    /**
     * This method loads the categories from the database into a combo box
     */
    private void loadCategories() {
        //create and execute the SQL string to get num of records and get the RecordSet
        CG41App.dbObject.sqlString = "select COUNT(*) from category";
        thisError = CG41App.dbObject.getCountBySelect();
        numRecords = CG41App.dbObject.NumberOfRecords;
        CG41App.dbObject.sqlString = "select CatID, CategoryName from category";
        thisError = CG41App.dbObject.getRecordSetBySelect();
        if (thisError == 0) {
            try {
                for (i = 0; i < numRecords; i++) {
                    //for each item in the RS, add the category
                    CG41App.dbObject.rs.next();
                    category = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2);
                    catCombox.addItem(category);
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
}
```

```

@SuppressWarnings ("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents () {

    jPanel1 = new javax.swing.JPanel ();
    catIDOutLbl = new javax.swing.JLabel ();
    loadCatBtn = new javax.swing.JButton ();
    catNameLbl = new javax.swing.JLabel ();
    editCatBtn = new javax.swing.JButton ();
    catIdLbl = new javax.swing.JLabel ();
    catNamFld = new javax.swing.JTextField ();
    exitBtn = new javax.swing.JButton ();
    catCombox = new javax.swing.JComboBox ();
    catLbl = new javax.swing.JLabel ();

    setDefaultCloseOperation (javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance (cg41.CG41App.class).getContext ().getResourceMap (EditCategory.
    class);
    setTitle (resourceMap.getString ("Form.title")); // NOI18N
    setName ("Form"); // NOI18N

    jPanel1.setBorder (javax.swing.BorderFactory.createTitledBorder (resourceMap.getString (
    "jPanel1.border.title"))); // NOI18N
    jPanel1.setName ("jPanel1"); // NOI18N

    catIDOutLbl.setText (resourceMap.getString ("catIDOutLbl.text")); // NOI18N
    catIDOutLbl.setName ("catIDOutLbl"); // NOI18N

    loadCatBtn.setText (resourceMap.getString ("loadCatBtn.text")); // NOI18N
    loadCatBtn.setName ("loadCatBtn"); // NOI18N
    loadCatBtn.addActionListener (new java.awt.event.ActionListener () {
        public void actionPerformed (java.awt.event.ActionEvent evt) {
            loadCatBtnActionPerformed (evt);
        }
    });

    catNameLbl.setText (resourceMap.getString ("catNameLbl.text")); // NOI18N
    catNameLbl.setName ("catNameLbl"); // NOI18N

    editCatBtn.setText (resourceMap.getString ("editCatBtn.text")); // NOI18N
    editCatBtn.setName ("editCatBtn"); // NOI18N
    editCatBtn.addActionListener (new java.awt.event.ActionListener () {
        public void actionPerformed (java.awt.event.ActionEvent evt) {
            editCatBtnActionPerformed (evt);
        }
    });

    catIdLbl.setText (resourceMap.getString ("catIdLbl.text")); // NOI18N
    catIdLbl.setName ("catIdLbl"); // NOI18N

    catNamFld.setText (resourceMap.getString ("catNamFld.text")); // NOI18N
    catNamFld.setName ("catNamFld"); // NOI18N

    exitBtn.setText (resourceMap.getString ("exitBtn.text")); // NOI18N
    exitBtn.setName ("exitBtn"); // NOI18N
    exitBtn.addActionListener (new java.awt.event.ActionListener () {

```

```

        public void actionPerformed(java.awt.event.ActionEvent evt) {
            exitBtnActionPerformed(evt);
        }
    });

    catCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
    catCombox.setName("catCombox"); // NOI18N

    catLbl.setText(resourceMap.getString("catLbl.text")); // NOI18N
    catLbl.setName("catLbl"); // NOI18N

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(10, 10, 10)
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(catNameLbl)
                        .addComponent(catLbl))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(catCombox, javax.swing.GroupLayout.PREFERRED_SIZE, 209, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(loadCatBtn, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                    .addGroup(jPanel1Layout.createSequentialGroup()
                        .addComponent(catNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, 137, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(38, 38, 38)
                        .addComponent(catIdLbl)
                        .addComponent(catIDOutLbl, javax.swing.GroupLayout.DEFAULT_SIZE, 86, Short.MAX_VALUE)))
                .addContainerGap(10, true))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(editCatBtn)
                .addGap(10, 10, 10)
                .addComponent(exitBtn))
            .addGap(10, 10, 10)
    );
    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(10, 10, 10)
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(catNameLbl)
                    .addComponent(catLbl)
                    .addComponent(catCombox)
                    .addComponent(loadCatBtn)
                    .addComponent(catNamFld)
                    .addComponent(catIdLbl)
                    .addComponent(catIDOutLbl)
                    .addComponent(editCatBtn)
                    .addComponent(exitBtn))
                .addContainerGap(10, true))
    );

```

```

        .addComponent(loadCatBtn)
        .addComponent(catCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
            swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .TRAILING)
            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.
                Alignment.BASELINE)
                .addComponent(catNameLbl)
                .addComponent(catNamFld, javax.swing.GroupLayout.PREFERRED_SIZE,
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                        PREFERRED_SIZE))
            .addGroup(jPanellLayout.createSequentialGroup())
            .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.
                Alignment.BASELINE)
                .addComponent(catIdLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
                    17, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(catIDOutLbl))
            .addGap(5, 5, 5)))
        .addGap(21, 21, 21)
        .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(exitBtn)
            .addComponent(editCatBtn))
        .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

```

```

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
    );

```

```

layout.setVerticalGroup(
    layout.createSequentialGroup()
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
    );

```

```

jPanell.getAccessibleContext().setAccessibleName(resourceMap.getString(
    "jPanell.AccessibleContext.accessibleName")); // NOI18N

```

```
pack();
```

```
    } // </editor-fold> // GEN-END: initComponents
```

```
/**
```

```

 * This method waits for the user to press the edit button and then amends the
 * category in the database with the edited version
 */

```

```

private void editCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_editCatBtnActionPerformed

```



```

myLocalError = false; //set all variables to null
thisError = 0;
myLocalString = "";

// Check if all fields are filled in. If not, then print an error
// message and don't do any more processing
if (catNamFld.getText().equals("")) {
    myLocalError = true;

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Please Enter All Fields.");
    MyMessage.setTitle("Error Editing Category");
    MyMessage.setVisible(true);
}
errorMessage = "Please check the following: \n\n";
if (!myLocalError) {
    if (!CG41App.validateObject.validateCategoryName(catNamFld.getText())) {
        myLocalError = true;
        errorMessage = errorMessage + "Category name should begin with a letter"
            + " and be no longer than 20 characters and can include"
            + " spaces, dashes and apostrophes. \n\n";
    }
}
if (!myLocalError) {
    // TODO Check if already in database
}
if (!myLocalError) {
    myLocalString = "update category set "
        + "CategoryName=" + "'" + catNamFld.getText().toString() + "'"
        + " where CatID= " + catIDOutLbl.getText();
    CG41App.dbObject.sqlString = myLocalString;
    thisError = CG41App.dbObject.updateRecord();
}
if (0 == thisError) { //set the new governor ID so the user can see it
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Category was successfully edited.");
    MyMessage.setTitle("Category Edited Successfully");
    MyMessage.setVisible(true);
}
} //GEN-LAST:event_editCatBtnActionPerformed

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} //GEN-LAST:event_exitBtnActionPerformed

/**
 * This method loads the selected category from the combo box into the fields
 * for editing
 */
private void loadCatBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_loadCatBtnActionPerformed
    myLocalString = ""; //reset variables

```

```

myCat = "";
thisError = 0;

myCat = catCombox.getSelectedItem().toString(); //get category string
myCatArray = myCat.split(","); //split it to make it handlable
myLocalString = "select * from category where CatID = '"
    + myCatArray[0] + "'"; //select the category where the IDs match
CG41App.dbObject.sqlString = myLocalString; //set the SQL string
thisError = CG41App.dbObject.getRecordSetBySelect(); //execute the query
try {
    CG41App.dbObject.rs.next();
    //output the category name for editing as well as the category ID for later
    catNamFld.setText(CG41App.dbObject.rs.getString("CategoryName"));
    catIDOutLbl.setText((myCatArray[0]));
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    thisError = ex.getErrorCode();
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
} //GEN-LAST:event_loadCatBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
            getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(EditCategory.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(EditCategory.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(EditCategory.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(EditCategory.class.getName()).log(java.util.

```

```
        logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new EditCategory().setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox catCombox;
private javax.swing.JLabel catIDOutLbl;
private javax.swing.JLabel catIdLbl;
private javax.swing.JLabel catLbl;
private javax.swing.JTextField catNamFld;
private javax.swing.JLabel catNameLbl;
private javax.swing.JButton editCatBtn;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton loadCatBtn;
// End of variables declaration//GEN-END:variables
}
```

```

package cg41;

import java.sql.SQLException;
import javax.swing.WindowConstants;

public class EditCourse extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString, myLocalString2 = null;
    String course, myCourse, catName, errorMessage, category, myCat = "";
    String myCourseArray[], myCategoryArray[];
    int thisError, thisError2, i, numRecords, catID = 0;

    /** Creates new form EditCourse */
    public EditCourse() {
        initComponents();
        loadCourses();
    }

    /**
     * This method loads the courses into a combo box for the user to select
     * from
     */
    private void loadCourses() {
        CG41App.dbObject.sqlString = "select COUNT(*) from course"; //set sql query
        thisError = CG41App.dbObject.getCountBySelect(); //execute query
        numRecords = CG41App.dbObject.NumberOfRecords; //get number of records
        //set and execute a search query
        CG41App.dbObject.sqlString = "select CourseID, CatID, CourseName from course";
        thisError = CG41App.dbObject.getRecordSetBySelect();
        if (thisError == 0) {
            try {
                //for each number of records add each course to the combo box
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next();
                    course = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(3)
                        + ", " + getCategoryName(Integer.parseInt(CG41App.dbObject.rs.
                            getString(2)));
                    courseComboBox.addItem(course);
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /**
     * This method is for loading the categories into the combo box for the

```

```

    * user to select when the course is edited
    */
private void loadCategories() {
    /**set SQL queries for counting the amount of records and then selecting
    * the records from the database
    */
    CG41App.dbObject.sqlString = "select COUNT(*) from category";
    thisError = CG41App.dbObject.getCountBySelect();
    numRecords = CG41App.dbObject.NumberOfRecords;
    CG41App.dbObject.sqlString = "select CatID, CategoryName from category";
    thisError = CG41App.dbObject.getRecordSetBySelect();
    if (thisError == 0) {
        try {
            //for the amount of records found, add them to the combo box
            for (i = 0; i < numRecords; i++) {
                CG41App.dbObject.rs.next();
                category = CG41App.dbObject.rs.getString(1) + ", "
                    + CG41App.dbObject.rs.getString(2);
                catEditCombox.addItem(category);
            }
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();

            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
    }
}

/**
 * This method will take an integer ID for the category as an input
 * and will return the name of the category associated with the ID as a
 * string
 */
public String getCategoryName(int catID) {
    myLocalString2 = ""; //rest variables
    thisError2 = 0;
    myLocalString2 = "select * from category where CatID = '"
        + catID + "'"; //set SQL statement for searching
    CG41App.dbObject2.sqlString = myLocalString2;
    thisError2 = CG41App.dbObject2.getRecordSetBySelect(); //select the record I need
    try { //get the category name and prepare it for the return
        CG41App.dbObject2.rs.next();
        catName = CG41App.dbObject2.rs.getString("CategoryName");
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        thisError2 = ex.getErrorCode();
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
    }
}

```

```

        MyMessage.setVisible(true);
    }
    return catName;
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    editCourseBtn = new javax.swing.JButton();
    courseNamEditFld = new javax.swing.JTextField();
    exitBtn = new javax.swing.JButton();
    courseCombox = new javax.swing.JComboBox();
    loadCourseBtn = new javax.swing.JButton();
    catIDLbl = new javax.swing.JLabel();
    courseIDOutLbl = new javax.swing.JLabel();
    courseLbl = new javax.swing.JLabel();
    courseNameLbl = new javax.swing.JLabel();
    catEditCombox = new javax.swing.JComboBox();
    catLbl = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(EditCourse.
    class);
    setTitle(resourceMap.getString("Form.title")); // NOI18N
    setName("Form"); // NOI18N

    jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
    "jPanel1.border.title"))); // NOI18N
    jPanel1.setName("jPanel1"); // NOI18N

    editCourseBtn.setText(resourceMap.getString("editCourseBtn.text")); // NOI18N
    editCourseBtn.setName("editCourseBtn"); // NOI18N
    editCourseBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            editCourseBtnActionPerformed(evt);
        }
    });

    courseNamEditFld.setName("courseNamEditFld"); // NOI18N

    exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
    exitBtn.setName("exitBtn"); // NOI18N
    exitBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            exitBtnActionPerformed(evt);
        }
    });

    courseCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));

```

-4-

-5-


```

        .addContainerGap ()
        .addComponent (jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
        GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap ()
    );
    layout.setVerticalGroup (
        layout.createParallelGroup (javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup (layout.createSequentialGroup ()
            .addContainerGap ()
            .addComponent (jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap ()
        );

    jPanell.getAccessibleContext ().setAccessibleName (resourceMap.getString (
        "jPanell.AccessibleContext.accessibleName")); // NOI18N

    pack ();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method will wait for the edit button to be pressed and then
 * will collect all the information needed for the edit
 */
private void editCourseBtnActionPerformed (java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_editCourseBtnActionPerformed
    myLocalError = false; //rest the variables
    thisError = 0;
    myLocalString = "";

    // Check if all fields are filled in. If not, then print an error
    // message and don't do any more processing
    if (courseNamEditFld.getText ().equals ("")) {
        myLocalError = true;

        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setDefaultCloseOperation (WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage ("Please Enter All Fields.");
        MyMessage.setTitle ("Error Editing Course");
        MyMessage.setVisible (true);
    }

    if (catEditCombox.getSelectedItem ().toString ().equals ("-")) {
        myLocalError = true;

        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setDefaultCloseOperation (WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage ("Please Select a Category.");
        MyMessage.setTitle ("Error Editing Course");
        MyMessage.setVisible (true);
    }

    errorMessage = "Please check the following: \n\n";
    if (!myLocalError) {
        //validation of course name
        if (!CG41App.validateObject.validateCourseName (courseNamEditFld.getText ())) {
            myLocalError = true;

```

```

        errorMessage = errorMessage + "The course name should begin with"
            + "a letter and be followed by up to 20 letters, numbers"
            + "apostrophes, spaces and dashes.";
    }
    if (myLocalError) {
        //One of the validation checks failed
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage(errorMessage);
        MyMessage.setTitle("Error Editing Course");
        MyMessage.setVisible(true);
    }
}
if (!myLocalError) {
    // TODO Check if already in database
}
if (!myLocalError) { //this will collect the information and amend the course
    myCat = catEditCombox.getSelectedItem().toString();
    myCategoryArray = myCat.split(","); //get category name and split it
    catID = Integer.parseInt(myCategoryArray[0]); //get category ID
    myLocalString = "update course set " + "CourseName= '" //create string
        + courseNamEditFld.getText().toString() + "', CatID = '" + catID
        + "' where CourseID= " + courseIDOutLbl.getText();
    CG41App.dbObject.sqlString = myLocalString; //set SQL string
    thisError = CG41App.dbObject.insertRecord();//execute query
}
if (0 == thisError && !myLocalError) {
    //if edit is successful
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Course edited successfully");
    MyMessage.setTitle("Edit Successful");
    MyMessage.setVisible(true);
}
} //GEN-LAST:event_editCourseBtnActionPerformed
/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} //GEN-LAST:event_exitBtnActionPerformed
/**
 * This method waits for the load button to be pressed and will load the selected
 * course into the fields for editing
 */
private void loadCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_loadCourseBtnActionPerformed
    myLocalString = ""; //reset variables
    myCourse = "";
    thisError = 0;

    myCourse = courseCombox.getSelectedItem().toString(); //get course information
    myCourseArray = myCourse.split(","); //split it to get the ID
    myLocalString = "select * from course where CourseID = '"
        + myCourseArray[0] + "'"; //find the course where the ID matches
    CG41App.dbObject.sqlString = myLocalString; //set the SQL string

```

```

thisError = CG41App.dbObject.getRecordSetBySelect(); //execute
try { //get and set the course name as well as ID for later
    CG41App.dbObject.rs.next();
    courseNamEditFld.setText(CG41App.dbObject.rs.getString("CourseName"));
    courseIDOutLbl.setText((myCourseArray[0]));
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    thisError = ex.getErrorCode();
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
loadCategories(); //need to load the categories for the user to choose from
} //GEN-LAST:event_loadCourseBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(EditCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(EditCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(EditCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(EditCourse.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {

```

```
        new EditCourse().setVisible(true);
    }
});
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox catEditCombox;
private javax.swing.JLabel catIDLbl;
private javax.swing.JLabel catLbl;
private javax.swing.JComboBox courseCombox;
private javax.swing.JLabel courseIDOutLbl;
private javax.swing.JLabel courseLbl;
private javax.swing.JTextField courseNamEditFld;
private javax.swing.JLabel courseNameLbl;
private javax.swing.JButton editCourseBtn;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanell;
private javax.swing.JButton loadCourseBtn;
// End of variables declaration//GEN-END:variables
}
```

```

package cg41;

import java.sql.SQLException;
import java.util.Date;
import javax.swing.WindowConstants;

public class EditGovernor extends javax.swing.JFrame {

    boolean myLocalError = false;
    String myLocalString, endDateString, governor, myGov,
        elecDateString, errorMessage, myGovArray[];
    int thisError, i, numRecords = 0;
    Date endDate;
    Date elecDate;

    /** Creates new form AddGovernor */
    public EditGovernor() {
        initComponents();
        loadGovernors();
    }
    /**
     * This method loads each governor from the database into a combo box
     */
    private void loadGovernors() {
        CG41App.dbObject.sqlString = "select COUNT(*) from governor"; //count governors
        thisError = CG41App.dbObject.getCountBySelect(); //execute query
        numRecords = CG41App.dbObject.NumberOfRecords;
        CG41App.dbObject.sqlString = "select GovID, Firstname, Surname from governor";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
        if (thisError == 0) {
            try {
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next(); //concatenate each governor
                    governor = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2) + " "
                        + CG41App.dbObject.rs.getString(3);
                    compGovCombox.addItem(governor); //add the governor
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */

```

```
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    govIdOutLbl = new javax.swing.JLabel();
    endDateLbl = new javax.swing.JLabel();
    loadGovBtn = new javax.swing.JButton();
    endDateChooser = new com.toedter.calendar.JDateChooser();
    lstNamLbl = new javax.swing.JLabel();
    frstNamLbl = new javax.swing.JLabel();
    elecDateChooser = new com.toedter.calendar.JDateChooser();
    editGovBtn = new javax.swing.JButton();
    govIdLbl = new javax.swing.JLabel();
    lstNamFld = new javax.swing.JTextField();
    frstNamFld = new javax.swing.JTextField();
    elecDateLbl = new javax.swing.JLabel();
    exitBtn = new javax.swing.JButton();
    compGovCombox = new javax.swing.JComboBox();
    compGovLbl = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(EditGovernor.
    class);
    setTitle(resourceMap.getString("Form.title")); // NOI18N
    setName("Form"); // NOI18N

    jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
    "jPanel1.border.title"))); // NOI18N
    jPanel1.setName("jPanel1"); // NOI18N

    govIdOutLbl.setText(resourceMap.getString("govIdOutLbl.text")); // NOI18N
    govIdOutLbl.setName("govIdOutLbl"); // NOI18N

    endDateLbl.setText(resourceMap.getString("endDateLbl.text")); // NOI18N
    endDateLbl.setName("endDateLbl"); // NOI18N

    loadGovBtn.setText(resourceMap.getString("loadGovBtn.text")); // NOI18N
    loadGovBtn.setName("loadGovBtn"); // NOI18N
    loadGovBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            loadGovBtnActionPerformed(evt);
        }
    });

    endDateChooser.setName("endDateChooser"); // NOI18N

    lstNamLbl.setText(resourceMap.getString("lstNamLbl.text")); // NOI18N
    lstNamLbl.setName("lstNamLbl"); // NOI18N

    frstNamLbl.setText(resourceMap.getString("frstNamLbl.text")); // NOI18N
    frstNamLbl.setName("frstNamLbl"); // NOI18N

    elecDateChooser.setName("elecDateChooser"); // NOI18N

    editGovBtn.setText(resourceMap.getString("editGovBtn.text")); // NOI18N
```

-3-

```

        .addComponent(editGovBtn)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(exitBtn))
    .addComponent(endDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE, 149
, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addComponent(1stNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, 185,
javax.swing.GroupLayout.PREFERRED_SIZE)
    .addComponent(elecDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
149, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGroup(jPanell1Layout.createSequentialGroup())
        .addComponent(frstNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, 137
, javax.swing.GroupLayout.PREFERRED_SIZE)
        .addGap(38, 38, 38)
        .addComponent(govIdLbl)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
        .addComponent(govIdOutLbl, javax.swing.GroupLayout.DEFAULT_SIZE, 86,
Short.MAX_VALUE)))
    .addContainerGap())
);
jPanell1Layout.setVerticalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jPanell1Layout.createSequentialGroup())
        .addContainerGap()
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
            .addComponent(compGovLbl)
            .addComponent(loadGovBtn)
            .addComponent(compGovCombox, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
PREFERRED_SIZE))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.TRAILING)
            .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.
Alignment.BASELINE)
                .addComponent(frstNamLbl)
                .addComponent(frstNamFld, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
PREFERRED_SIZE))
            .addGroup(jPanell1Layout.createSequentialGroup())
                .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.
Alignment.BASELINE)
                    .addComponent(govIdLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
17, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(govIdOutLbl))
                .addGap(5, 5, 5)))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
            .addComponent(1stNamLbl)
            .addComponent(1stNamFld, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
        .addGap(18, 18, 18)
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.LEADING)
            .addComponent(elecDateLbl)
            .addComponent(elecDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,

```



```

        javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
        PREFERRED_SIZE))
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
    .LEADING)
        .addComponent(endDateLbl)
        .addComponent(endDateChooser, javax.swing.GroupLayout.PREFERRED_SIZE,
        javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
        PREFERRED_SIZE))
    .addGap(10, 10, 10)
    .addGroup(jPanellLayout.createParallelGroup(javax.swing.GroupLayout.Alignment
    .BASELINE)
        .addComponent(exitBtn)
        .addComponent(editGovBtn))
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
    );

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
        );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

    jPanell.getAccessibleContext().setAccessibleName(resourceMap.getString(
    "jPanell.AccessibleContext.accessibleName")); // NOI18N

    pack();
} // </editor-fold> // GEN-END: initComponents
/**
 * This method will wait for the edit button to be pressed before sending off
 * the edited governor to add to the DB
 */
private void editGovBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_editGovBtnActionPerformed
    myLocalError = false; //set used variables to blank
    thisError = 0;
    myLocalString = "";

    elecDate = elecDateChooser.getDate(); //get the dates and format them
    elecDateString = String.format("%1$tY-%1$tm-%1$td", elecDate);
    endDate = endDateChooser.getDate();
    endDateString = String.format("%1$tY-%1$tm-%1$td", endDate);

    // Check if all fields are filled in. If not, then print an error

```

```

// message and don't do any more processing
if (firstNamFld.getText().equals(""))
    || lstNamFld.getText().equals("")
    || elecDateString.equals("null-null-null")
    || endDateString.equals("null-null-null")) {
    myLocalError = true;
}
if (myLocalError) {
    // One field must be blank, so error
    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    MyMessage.setMessage("Please Enter All Fields.");
    MyMessage.setTitle("Error Adding Governor");
    MyMessage.setVisible(true);
}
errorMessage = "Please check the following: \n\n";
if (!myLocalError) {
    //validation of first name
    if (!CG41App.validateObject.validateFirstName(firstNamFld.getText())) {
        myLocalError = true;
        errorMessage = errorMessage + "First name should begin with a capital"
            + " and be no longer than 10 characters. \n\n";
    }
    //validation of last name
    if (!CG41App.validateObject.validateSurname(firstNamFld.getText())) {
        myLocalError = true;
        errorMessage = errorMessage + "Last name should begin with a letter"
            + " and be no longer than 20 characters and can include"
            + " spaces, dashes and apostrophes. \n\n";
    }
    //check to see if the election date is after the end date
    if (elecDate.after(endDate)) {
        myLocalError = true;
        errorMessage = errorMessage + "Make sure the elected date is before"
            + " the 'end date'.";
    }
    //DB check
    if (!myLocalError) {
        // TODO Check if already in database
    }
}
if (!myLocalError) { //as long as there is no error, we can continue
    myLocalString = "update governor set " + "Firstname=" + ""
        + firstNamFld.getText().toString() + ""
        + ", Surname=" + "" + lstNamFld.getText().toString() + ""
        + ", ElectionDate=" + "" + elecDateString + "" + ", EndDate="
        + "" + endDateString + "" + " where GovID= "
        + govIdOutLbl.getText();
    CG41App.dbObject.sqlString = myLocalString;
    thisError = CG41App.dbObject.updateRecord(); //execute string
    if (thisError == 0) {
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage("Editing the Governor was successful.");
        MyMessage.setTitle("Editing Governor Complete");
        MyMessage.setVisible(true);
    }
}

```

```

    } else if (myLocalError) {
        //One of the validation checks failed
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        MyMessage.setMessage(errorMessage);
        MyMessage.setTitle("Error Adding Governor");
        MyMessage.setVisible(true);
    }
} //GEN-LAST:event_editGovBtnActionPerformed

/**
 * This method closes the window when the exit button is pressed
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_exitBtnActionPerformed
        dispose();
    //GEN-LAST:event_exitBtnActionPerformed
}

/**
 * This method will load the selected governor from the governor combo box and
 * enter each part into its respective editable field for the user to edit
 */
private void loadGovBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_loadGovBtnActionPerformed
        myLocalString = ""; //set all used variables to null
        myGov = "";
        thisError = 0;
        myGov = compGovCombox.getSelectedItem().toString(); //get the selected governor
        myGovArray = myGov.split(","); //split it by comma
        myLocalString = "select * from governor where GovID = '"
            + myGovArray[0] + "'";
        CG41App.dbObject.sqlString = myLocalString; //set the MySQL string
        thisError = CG41App.dbObject.getRecordSetBySelect(); //get the error code
        try { //add the governor to the fields
            CG41App.dbObject.rs.next();
            //get each part of the governor from the database query
            frstNamFld.setText(CG41App.dbObject.rs.getString("Firstname"));
            lstNamFld.setText(CG41App.dbObject.rs.getString("Surname"));
            elecDateChooser.setDate(CG41App.dbObject.rs.getDate("ElectionDate"));
            endDateChooser.setDate(CG41App.dbObject.rs.getDate("EndDate"));
            govIdOutLbl.setText((myGovArray[0]));
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
    } //GEN-LAST:event_loadGovBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)

```

```

">
/* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
and feel.
 * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
 */
try {
    for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
        if ("Nimbus".equals(info.getName())) {
            javax.swing.UIManager.setLookAndFeel(info.getClassName());
            break;
        }
    }
} catch (ClassNotFoundException ex) {
    java.util.logging.Logger.getLogger(EditGovernor.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
} catch (InstantiationException ex) {
    java.util.logging.Logger.getLogger(EditGovernor.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
} catch (IllegalAccessException ex) {
    java.util.logging.Logger.getLogger(EditGovernor.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
} catch (javax.swing.UnsupportedLookAndFeelException ex) {
    java.util.logging.Logger.getLogger(EditGovernor.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
}
//</editor-fold>

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {

    public void run() {
        new EditGovernor().setVisible(true);
    }
});
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox compGovCombox;
private javax.swing.JLabel compGovLbl;
private javax.swing.JButton editGovBtn;
private com.toedter.calendar.JDateChooser elecDateChooser;
private javax.swing.JLabel elecDateLbl;
private com.toedter.calendar.JDateChooser endDateChooser;
private javax.swing.JLabel endDateLbl;
private javax.swing.JButton exitBtn;
private javax.swing.JTextField frstNamFld;
private javax.swing.JLabel frstNamLbl;
private javax.swing.JLabel govIdLbl;
private javax.swing.JLabel govIdOutLbl;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton loadGovBtn;
private javax.swing.JTextField lstNamFld;
private javax.swing.JLabel lstNamLbl;
// End of variables declaration//GEN-END:variables
}

```

```
package cg41;

import java.sql.SQLException;

public class RemoveCategory extends javax.swing.JFrame {

    int thisError, numRecords, i;
    String category, myLocalString, myCatArray[], myCat;
    boolean removeCategory;

    /** Creates new form RemoveCompletedCourse */
    public RemoveCategory() {
        initComponents();
        loadCategories();
    }

    /**
     * Method used to load the categories into the combo box
     */
    private void loadCategories() {
        CG41App.dbObject.sqlString = "select COUNT(*) from category"; //count categories
        thisError = CG41App.dbObject.getCountBySelect(); //execute
        numRecords = CG41App.dbObject.NumberOfRecords;
        CG41App.dbObject.sqlString = "select CatID, CategoryName from category";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
        if (thisError == 0) { //as long as there is no error
            try {
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next(); //concatenate a category string
                    category = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2);
                    categoryComboBox.addItem(category); //add it to the combobox
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
    }
}
```

```
categoryCombox = new javax.swing.JComboBox();
exitBtn = new javax.swing.JButton();
removeCategoryBtn = new javax.swing.JButton();
hintLbl = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
RemoveCategory.class);
setTitle(resourceMap.getString("Form.title")); // NOI18N
setName("Form"); // NOI18N

jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
"jPanel1.border.title"))); // NOI18N
jPanel1.setName("jPanel1"); // NOI18N

categoryCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
categoryCombox.setName("categoryCombox"); // NOI18N

exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
exitBtn.setName("exitBtn"); // NOI18N
exitBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exitBtnActionPerformed(evt);
    }
});

removeCategoryBtn.setText(resourceMap.getString("removeCategoryBtn.text")); // NOI18N
removeCategoryBtn.setName("removeCategoryBtn"); // NOI18N
removeCategoryBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        removeCategoryBtnActionPerformed(evt);
    }
});

hintLbl.setText(resourceMap.getString("hintLbl.text")); // NOI18N
hintLbl.setName("hintLbl"); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(removeCategoryBtn)
                .addComponent(exitBtn)
                .addComponent(categoryCombox, 0, 261, Short.MAX_VALUE)
                .addComponent(hintLbl))
            .addGap(10, 10, 10))
);
jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(removeCategoryBtn)
            .addPreferredGap(
```

```

        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel1Layout.
createSequentialGroup())
        .addComponent(hintLbl)
        .addGap(11, 11, 11)
        .addComponent(categoryCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 41,
Short.MAX_VALUE)
        .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
            .addComponent(exitBtn)
            .addComponent(removeCategoryBtn))
        .addContainerGap()
    );

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap()
            )
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addContainerGap()
            )
    );

    pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * Method used to close the window on button click
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST:event_exitBtnActionPerformed
    dispose();
} // GEN-LAST:event_exitBtnActionPerformed

/**
 * Removes the selected governor from the database on press of button
 */
private void removeCategoryBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST:event_removeCategoryBtnActionPerformed
    myLocalizedString = ""; // reset used variables
    myCat = "";
    thisError = 0;
    removeCategory = false;

    myCat = categoryCombox.getSelectedItem().toString(); // get category info
    if (myCat.equals("-")) { // check to make sure the user has selected a category

```

-4-


```
        public void run() {
            new RemoveCategory().setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox categoryCombox;
private javax.swing.JButton exitBtn;
private javax.swing.JLabel hintLbl;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton removeCategoryBtn;
// End of variables declaration//GEN-END:variables
}
```

```
package cg41;

import java.sql.SQLException;

public class RemoveCompletedCourse extends javax.swing.JFrame {

    int thisError, i, numRecords, courseID, thisError2;
    String governor, myLocalString, myGov, myGovArray[], myCompCourseArray[];
    String compCourseEntry, courseName, myCompCourse;
    boolean myLocalError;

    /** Creates new form RemoveCompletedCourse */
    public RemoveCompletedCourse() {
        initComponents();
        loadGovernors();
    }

    /**
     * Method used to load the governors into the combo box
     */
    private void loadGovernors() {
        CG41App.dbObject.sqlString = "select COUNT(*) from governor"; //count governors
        thisError = CG41App.dbObject.getCountBySelect(); //execute count
        numRecords = CG41App.dbObject.NumberOfRecords; //store the amount of governors
        //create a select statement to get all the governor records
        CG41App.dbObject.sqlString = "select GovID, Firstname, Surname from governor";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute select
        if (thisError == 0) { //if no error
            try {
                //for the amount of records, add the governors to the combo box
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next();
                    governor = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2) + " "
                        + CG41App.dbObject.rs.getString(3); //create governor string
                    compGovCombox.addItem(governor); //add governor
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
```

```
// <editor-fold defaultstate="collapsed" desc="Generated Code"> //GEN-BEGIN: initComponents
private void initComponents () {

    jPanel1 = new javax.swing.JPanel();
    compGovLbl = new javax.swing.JLabel();
    compGovCombox = new javax.swing.JComboBox();
    coursesByGovernor = new javax.swing.JLabel();
    compCourseCombox = new javax.swing.JComboBox();
    exitBtn = new javax.swing.JButton();
    removeCompCourseBtn = new javax.swing.JButton();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
    RemoveCompletedCourse.class);
    setTitle(resourceMap.getString("Form.title")); // NOI18N
    setName("Form"); // NOI18N

    jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
    "jPanel1.border.title"))); // NOI18N
    jPanel1.setName("jPanel1"); // NOI18N

    compGovLbl.setText(resourceMap.getString("compGovLbl.text")); // NOI18N
    compGovLbl.setName("compGovLbl"); // NOI18N

    compGovCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
    compGovCombox.setName("compGovCombox"); // NOI18N
    compGovCombox.addItemListener(new java.awt.event.ItemListener() {
        public void itemStateChanged(java.awt.event.ItemEvent evt) {
            compGovComboxItemStateChanged(evt);
        }
    });

    coursesByGovernor.setText(resourceMap.getString("coursesByGovernor.text")); // NOI18N
    coursesByGovernor.setName("coursesByGovernor"); // NOI18N

    compCourseCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
    compCourseCombox.setName("compCourseCombox"); // NOI18N

    exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
    exitBtn.setName("exitBtn"); // NOI18N
    exitBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            exitBtnActionPerformed(evt);
        }
    });

    removeCompCourseBtn.setText(resourceMap.getString("removeCompCourseBtn.text")); //
    NOI18N
    removeCompCourseBtn.setName("removeCompCourseBtn"); // NOI18N
    removeCompCourseBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            removeCompCourseBtnActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
```

```

jPanell1.setLayout(jPanell1Layout);
jPanell1Layout.setHorizontalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanell1Layout.createSequentialGroup())
            .addContainerGap()
            .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
                .TRAILING)
                .addGroup(jPanell1Layout.createSequentialGroup())
                    .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.
                        Alignment.LEADING)
                        .addComponent(coursesByGovernor)
                        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.
                            GroupLayout.Alignment.TRAILING, false)
                                .addComponent(compGovCombox, javax.swing.GroupLayout.
                                    Alignment.LEADING, 0, javax.swing.GroupLayout.DEFAULT_SIZE,
                                    Short.MAX_VALUE)
                                .addComponent(compCourseCombox, javax.swing.GroupLayout.
                                    Alignment.LEADING, 0, 230, Short.MAX_VALUE))
                            .addComponent(compGovLbl, javax.swing.GroupLayout.PREFERRED_SIZE,
                                59, javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addGap(89, 89, 89))
                    .addGroup(jPanell1Layout.createSequentialGroup())
                        .addComponent(removeCompCourseBtn)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED
                            )))
            .addComponent(exitBtn)
            .addContainerGap()
);

jPanell1Layout.setVerticalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanell1Layout.createSequentialGroup())
            .addComponent(compGovLbl)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(compGovCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
                swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addComponent(coursesByGovernor)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(compCourseCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax
                .swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(57, 57, 57)
            .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
                .BASELINE)
                .addComponent(exitBtn)
                .addComponent(removeCompCourseBtn))
            .addContainerGap()
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())
            .addContainerGap()
            .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap()
);

```

```

    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup())
            .addContainerGap()
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap()
        );

    pack();
} // </editor-fold> // GEN-END: initComponents
/**
 * Method used to close the window on button click
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_exitBtnActionPerformed
    dispose();
} // GEN-LAST: event_exitBtnActionPerformed
/**
 * When the remove button is pressed, it will proceed to remove the completed
 * course selected from the database
 */
private void removeCompCourseBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_removeCompCourseBtnActionPerformed
    myLocalString = ""; // reset used variables
    myGov = "";
    thisError = 0;
    boolean removeCompCourse = false;
    // check to see if the user has selected a governor
    if (compGovCombox.getSelectedItem().toString().equals("-")) {
        removeCompCourse = false;
        // show error
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setTitle("Error");
        MyMessage.setMessage("Please select a governor to search by.");
        MyMessage.setVisible(true);
    } // if they've selected a governor, but not a course then
    else if (compCourseCombox.getSelectedItem().toString().equals("-")) {
        removeCompCourse = false;
        // show error
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setTitle("Error");
        MyMessage.setMessage("Please select a completed course to delete.");
        MyMessage.setVisible(true);
    } else {
        // set up a confirmation box to ensure the user wants to continue
        UserConfirmationBox userConfirm = new UserConfirmationBox(this, true);
        userConfirm.setMessage("Removing Completed Course:\n"
            + "The completed course that you wish to\nremove is:\n"
            + compCourseCombox.getSelectedItem().toString()
            + ".\nThis course was completed by:\n"
            + compGovCombox.getSelectedItem().toString()
            + ".\nIf you delete this completed course,\n"
            + "nothing else will be deleted.\n"
            + "Are you sure you want to continue?");
        userConfirm.setVisible(true);
    }
}

```

```

        removeCompCourse = userConfirm.removeConfirmation; //take the result from the
        confirm window
        userConfirm.dispose(); //make sure we close it
        if(removeCompCourse == false) { //if they don't want to delete
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("The Completed Course:\n"
                + compCourseCombox.getSelectedItem().toString()
                + ".\nWas not removed.");
            MyMessage.setVisible(true);
        }
    }
    if (removeCompCourse == true) { //if they confirm they want to delete
        myCompCourse = compCourseCombox.getSelectedItem().toString(); //get Completed
        Course info
        myCompCourseArray = myCompCourse.split(","); //split the completed course
        myLocalString = "delete from coursetaken where TakenID=" + myCompCourseArray[0];
        CG41App.dbObject2.sqlString = myLocalString; //set the SQL string
        thisError = CG41App.dbObject2.deleteRecord(); //execute the query

        //set the selected index of each combo box to dash
        compGovCombox.setSelectedIndex(0);
        compCourseCombox.setSelectedIndex(0);

        // Report completion to the user
        if(thisError == 0){
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("Completed Course Successfully Deleted");
            MyMessage.setVisible(true);
        }
        else{
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("There was an Error deleting the completed course.");
            MyMessage.setVisible(true);
        }
    }
} //GEN-LAST:event_removeCompCourseBtnActionPerformed
/**
 * As the governor combo box changes state by the user's control, the
 * completed course combo box should load the courses completed by the
 * selected governor
 */
private void compGovComboxItemStateChanged(java.awt.event.ItemEvent evt) {
    //GEN-FIRST:event_compGovComboxItemStateChanged
    myLocalString = ""; //rest variables
    myGov = "";
    thisError = 0;

    //reset the complete course combo box
    compCourseCombox.removeAllItems();
    compCourseCombox.addItem("-");

    myGov = compGovCombox.getSelectedItem().toString(); //get Gov info
    myGovArray = myGov.split(","); //split it to get the id
    myLocalString = "select COUNT(*) from coursetaken where GovID= '"
        + myGovArray[0] + "'"; //create mysql query
    CG41App.dbObject.sqlString = myLocalString; //set SQL string
    thisError = CG41App.dbObject.getCountBySelect(); //execute count

```

```

numRecords = CG41App.dbObject.NumberOfRecords; //save number of records

//Create SQL statement to find all the courses taken where only the
//governor ID appears
CG41App.dbObject.sqlString =
    "select TakenID, GovID, CourseID, DateTaken from coursetaken"
    + " where GovID= '" + myGovArray[0] + "'";
// Execute the request
thisError = CG41App.dbObject.getRecordSetBySelect();
if (thisError == 0) { //if no error, continue
    try {
        while (CG41App.dbObject.rs.next()) { // For each record
            courseID = CG41App.dbObject.rs.getInt("CourseID");
            //where getCourseName is called, it will return the course name of
            the taken course
            compCourseEntry = CG41App.dbObject.rs.getString("TakenID") + ", "
                //+ CG41App.dbObject.rs.getString("GovID") + ", "
                + getCourseName(courseID) + ", "
                + CG41App.dbObject.rs.getString("DateTaken");
            compCourseCombox.addItem(compCourseEntry); //add item
        }
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        thisError = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
}
} //GEN-LAST:event_compGovComboxItemStateChanged

/**
 * getCourseName will take a given courseID and find the course name
 * linked to the course ID form the database and return it as a string
 */
public String getCourseName(int courseInID) {
    myLocalString = ""; //reset variables
    myLocalError = false;
    thisError2 = 0;
    //set select SQL statement
    myLocalString = "select * from course where CourseID = '"
        + courseInID + "'";
    CG41App.dbObject2.sqlString = myLocalString; //set the string
    thisError2 = CG41App.dbObject2.getRecordSetBySelect(); //execute query
    try { //get the course name from the record set
        CG41App.dbObject2.rs.next();
        courseName = CG41App.dbObject2.rs.getString("CourseName");
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
    }
}

```

```
        thisError2 = ex.getErrorCode();
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
    return courseName;
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new RemoveCompletedCourse().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox compCourseCombox;
private javax.swing.JComboBox compGovCombox;
private javax.swing.JLabel compGovLbl;
private javax.swing.JLabel coursesByGovernor;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton removeCompCourseBtn;
// End of variables declaration//GEN-END:variables
}
```



```

package cg41;

import java.sql.SQLException;

public class RemoveCourse extends javax.swing.JFrame {
    int thisError, numRecords, i;
    String myCourse, myLocalString2, catName, myLocalString,
        myCourseArray[], course;
    boolean removeCourse;

    /** Creates new form RemoveCompletedCourse */
    public RemoveCourse() {
        initComponents();
        loadCourses();
    }

    public String getCategoryName(int catID){
        myLocalString2 = "";
        thisError = 0;
        myLocalString2 = "select * from category where CatID = '"
            + catID + "'";
        CG41App.dbObject2.sqlString = myLocalString2;
        thisError = CG41App.dbObject2.getRecordSetBySelect();
        try{
            CG41App.dbObject2.rs.next();
            catName = CG41App.dbObject2.rs.getString("CategoryName");
        }catch (SQLException ex){
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
        return catName;
    }

    /**
     * Method used to load the courses into the combo box
     */
    private void loadCourses() {
        //get the number of courses in the database to use later
        CG41App.dbObject.sqlString = "select COUNT(*) from course";
        thisError = CG41App.dbObject.getCountBySelect(); //execute
        numRecords = CG41App.dbObject.NumberOfRecords;
        //select all the courses to load from the database
        CG41App.dbObject.sqlString = "select CourseID, CatID, CourseName from course";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute
        if (thisError == 0) { //if no error
            try {
                //for amount of records, load each record in the combobox
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next();
                    //where getCategoryName is called, it will get the category
                    //name for the course recieved from the database
                }
            }
        }
    }
}

```

```

        course = CG41App.dbObject.rs.getString(1) + ", " +
+ getCategoryName(Integer.parseInt(CG41App.dbObject.rs.getString(2))) +
        ", " + CG41App.dbObject.rs.getString(3);
        courseCombox.addItem(course);
    }
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    thisError = ex.getErrorCode();

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    exitBtn = new javax.swing.JButton();
    removeCourseBtn = new javax.swing.JButton();
    courseCombox = new javax.swing.JComboBox();
    jLabel1 = new javax.swing.JLabel();
    jLabel2 = new javax.swing.JLabel();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
    Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(RemoveCourse.
    class);
    setTitle(resourceMap.getString("Form.title")); // NOI18N
    setName("Form"); // NOI18N

    jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
    "jPanel1.border.title"))); // NOI18N
    jPanel1.setName("jPanel1"); // NOI18N

    exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
    exitBtn.setName("exitBtn"); // NOI18N
    exitBtn.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            exitBtnActionPerformed(evt);
        }
    });

    removeCourseBtn.setText(resourceMap.getString("removeCourseBtn.text")); // NOI18N
    removeCourseBtn.setName("removeCourseBtn"); // NOI18N
    removeCourseBtn.addActionListener(new java.awt.event.ActionListener() {

```

```

        public void actionPerformed(java.awt.event.ActionEvent evt) {
            removeCourseBtnActionPerformed(evt);
        }
    });

    courseCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
    courseCombox.setName("courseCombox"); // NOI18N

    jLabel1.setText(resourceMap.getString("jLabel1.text")); // NOI18N
    jLabel1.setName("jLabel1"); // NOI18N

    jLabel2.setText(resourceMap.getString("jLabel2.text")); // NOI18N
    jLabel2.setName("jLabel2"); // NOI18N

    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(removeCourseBtn)
                    .addGap(18, 18, 18)
                    .addComponent(exitBtn)
                    .addGap(113, 113, 113))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(54, 54, 54)
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(courseCombox, 0, 260, Short.MAX_VALUE)
                    .addComponent(jLabel1))
                .addContainerGap())
    );

    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(courseCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 34, Short.MAX_VALUE)
                .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(exitBtn)
                    .addComponent(removeCourseBtn))
                .addContainerGap())
    );

```

```

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout (getContentPane ());
    getContentPane ().setLayout (layout);
    layout.setHorizontalGroup (
        layout.createParallelGroup (javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup (layout.createSequentialGroup ()
            .addGap ()
            .addComponent (jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGap ())
    );
    layout.setVerticalGroup (
        layout.createParallelGroup (javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup (layout.createSequentialGroup ()
            .addGap ()
            .addComponent (jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGap ())
    );

    pack ();
} // </editor-fold> // GEN-END: initComponents

/**
 * Method used to close the window on button click
 */
private void exitBtnActionPerformed (java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_exitBtnActionPerformed
    dispose ();
} // GEN-LAST: event_exitBtnActionPerformed

/**
 * Removes the selected governor from the database on press of button
 */
private void removeCourseBtnActionPerformed (java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_removeCourseBtnActionPerformed
    myLocalString = ""; // reset used variables
    myCourse = "";
    thisError = 0;
    removeCourse = false;

    myCourse = courseCombox.getSelectedItem ().toString (); // get Completed Course info
    if (myCourse.equals ("-")) {
        removeCourse = false;
        // show an error to the user
        UserMessageBox MyMessage = new UserMessageBox ();
        MyMessage.setMessage ("Please select a course to remove.");
        MyMessage.setVisible (true);
    } else {
        UserConfirmationBox userConfirm = new UserConfirmationBox (this, true);
        // make sure the user knows what they're removing
        userConfirm.setMessage ("Removing Course:\n"
            + "The course that you wish to remove is:\n"
            + myCourse
            + ".\nUpon deleting this course, \nany completed"
            + "\ncourses that require this course\nwill also be"
            + "deleted.\nAre you sure you want to continue?");
    }
}

```

```

        userConfirm.setVisible(true);
        removeCourse = userConfirm.removeConfirmation; //take the result from the
        confirm window
        userConfirm.dispose(); //make sure we close it
    }
    if (removeCourse == false) { //if they don't want to delete
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("The Course:\n"
            + myCourse
            + ".\nWas not removed.");
        MyMessage.setVisible(true);
    }
    else if(removeCourse == true) { //if they want to remove the course
        myCourseArray = myCourse.split(","); //split it to get the ID
        //delete where the course IDs match
        myLocalString = "delete from course where CourseID=" + myCourseArray[0];
        CG41App.dbObject2.sqlString = myLocalString; //set the SQL string
        thisError = CG41App.dbObject2.deleteRecord(); //execute the query

        //reset the UI for the user to remove another course
        courseCombox.removeAllItems();
        courseCombox.addItem("-");
        courseCombox.setSelectedIndex(0);
        loadCourses();

        // Report completiton to the user
        if(thisError == 0){
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("Course Successfully Deleted");
            MyMessage.setVisible(true);
        }else{
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("There was an Error deleting the Course.");
            MyMessage.setVisible(true);
        }
    }
}

} //GEN-LAST:event_removeCourseBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new RemoveCourse().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox courseCombox;
private javax.swing.JButton exitBtn;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton removeCourseBtn;

```

```
// End of variables declaration//GEN-END:variables
```

```
}
```

```

package cg41;

import java.sql.SQLException;

public class RemoveGovernor extends javax.swing.JFrame {

    int thisError, numRecords, i;
    String governor, myLocalString, myGovArray[], myGov;
    boolean removeGovernor;

    /** Creates new form RemoveCompletedCourse */
    public RemoveGovernor() {
        initComponents();
        loadGovernors();
    }

    /**
     * Method used to load the governors into the combo box
     */
    private void loadGovernors() {
        CG41App.dbObject.sqlString = "select COUNT(*) from governor"; //count governors
        thisError = CG41App.dbObject.getCountBySelect(); //execute
        numRecords = CG41App.dbObject.NumberOfRecords;
        CG41App.dbObject.sqlString = "select GovID, Firstname, Surname from governor";
        thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
        if (thisError == 0) { //as long as there is no error
            try {
                for (i = 0; i < numRecords; i++) {
                    CG41App.dbObject.rs.next(); //concatenate a governor string
                    governor = CG41App.dbObject.rs.getString(1) + ", "
                        + CG41App.dbObject.rs.getString(2) + " "
                        + CG41App.dbObject.rs.getString(3);
                    governorCombox.addItem(governor); //add it to the combobox
                }
            } catch (SQLException ex) {
                // Show errors in console
                System.out.println("SQLException: " + ex.getMessage());
                System.out.println("SQLState: " + ex.getSQLState());
                System.out.println("VendorError: " + ex.getErrorCode());
                thisError = ex.getErrorCode();

                UserMessageBox MyMessage = new UserMessageBox();
                MyMessage.setMessage("SQLException: " + ex.getMessage());
                MyMessage.setVisible(true);
            }
        }
    }

    /** This method is called from within the constructor to
     * initialise the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

```

```
jPanell1 = new javax.swing.JPanel();
governorCombox = new javax.swing.JComboBox();
exitBtn = new javax.swing.JButton();
removeGovernorBtn = new javax.swing.JButton();
hintLbl = new javax.swing.JLabel();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
RemoveGovernor.class);
setTitle(resourceMap.getString("Form.title")); // NOI18N
setName("Form"); // NOI18N

jPanell1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
"jPanell1.border.title"))); // NOI18N
jPanell1.setName("jPanell1"); // NOI18N

governorCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
governorCombox.setName("governorCombox"); // NOI18N

exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
exitBtn.setName("exitBtn"); // NOI18N
exitBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        exitBtnActionPerformed(evt);
    }
});

removeGovernorBtn.setText(resourceMap.getString("removeGovernorBtn.text")); // NOI18N
removeGovernorBtn.setName("removeGovernorBtn"); // NOI18N
removeGovernorBtn.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        removeGovernorBtnActionPerformed(evt);
    }
});

hintLbl.setText(resourceMap.getString("hintLbl.text")); // NOI18N
hintLbl.setName("hintLbl"); // NOI18N

javax.swing.GroupLayout jPanell1Layout = new javax.swing.GroupLayout(jPanell1);
jPanell1.setLayout(jPanell1Layout);
jPanell1Layout.setHorizontalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanell1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(removeGovernorBtn)
                .addComponent(exitBtn)
                .addComponent(governorCombox, 0, 289, Short.MAX_VALUE)
                .addComponent(hintLbl))
            .addGap(10, 10, 10))
);
jPanell1Layout.setVerticalGroup(
```



```

jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanell1Layout.
createSequentialGroup())
.addComponent(hintLbl)
.addGap(7, 7, 7)
.addComponent(governorCombox, javax.swing.GroupLayout.PREFERRED_SIZE, javax.
swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 43,
Short.MAX_VALUE)
.addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
.BASELINE)
.addComponent(exitBtn)
.addComponent(removeGovernorBtn))
.addContainerGap())
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
.addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addContainerGap())
);
layout.setVerticalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
.addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addContainerGap())
);

pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * Method used to close the window on button click
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_exitBtnActionPerformed
dispose();
} // GEN-LAST:event_exitBtnActionPerformed

/**
 * Removes the selected governor from the database on press of button
 */
private void removeGovernorBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_removeGovernorBtnActionPerformed
removeGovernor = false;
//Create a user confirmation box with a message to ensure that
//the user wants to delete the selected governor
if (governorCombox.getSelectedItem().toString().equals("-")) {
removeGovernor = false;

```

```

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setTitle("Error");
    MyMessage.setMessage("Please select a governor to remove.");
    MyMessage.setVisible(true);
} else {
    //set up a confirmation box to ensure the user wants to continue
    UserConfirmationBox userConfirm = new UserConfirmationBox(this, true);
    userConfirm.setMessage("Removing Governor:\n"
        + "The governor that you wish to remove is:\n"
        + governorCombox.getSelectedItem().toString()
        + ".\nUpon deleting this governor,\nany completed courses they have"
        + " completed\nwill also be removed.\nAre you sure you want to con"
        + "tinue?");
    userConfirm.setVisible(true);
    removeGovernor = userConfirm.removeConfirmation; //take the result from the
    confirm window
    userConfirm.dispose(); //make sure we close it
    if(removeGovernor == false) { //if they don't want to delete
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("The Governor:\n"
            + governorCombox.getSelectedItem().toString()
            + ".\nWas not removed.");
        MyMessage.setVisible(true);
    }
}
if (removeGovernor == true) { //if they confirm they want to delete
    myLocalString = ""; //set all used variables to null
    myGov = "";
    thisError = 0;

    myGov = governorCombox.getSelectedItem().toString(); //get Completed Course info
    myGovArray = myGov.split(","); //split the string by commas
    myLocalString = "delete from governor where GovID=" + myGovArray[0];
    CG41App.dbObject.sqlString = myLocalString; //set MySQL string
    thisError = CG41App.dbObject.deleteRecord(); //execute MySQL String

    governorCombox.removeAllItems(); //reset the goverors after removing
    governorCombox.addItem("-"); //the gover ready to remove another one
    governorCombox.setSelectedIndex(0);
    loadGovernors();

    // Report completion to the user
    if (thisError == 0) {
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setTitle("Governor Removed");
        MyMessage.setMessage("The Governor:\n" + myGov + ",\nwas successfully
        deleted");
        MyMessage.setVisible(true);
    } else {
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setTitle("Governor Remove Unsuccessful");
        MyMessage.setMessage("There was an Error deleting the Governor.");
        MyMessage.setVisible(true);
    }
}
}
} //GEN-LAST:event_removeGovernorBtnActionPerformed

```

```
/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new RemoveGovernor().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton exitBtn;
private javax.swing.JComboBox governorCombox;
private javax.swing.JLabel hintLbl;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton removeGovernorBtn;
// End of variables declaration//GEN-END:variables
}
```

```
package cg41;

import java.sql.SQLException;
import javax.swing.JTable;

public class ViewCategories extends javax.swing.JFrame {

    String catID, catName;
    int thisError, numRecords;

    public ViewCategories() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        refreshBtn = new javax.swing.JButton();
        exitBtn = new javax.swing.JButton();
        categoryScrlPane = new javax.swing.JScrollPane();
        categoryTbl = new javax.swing.JTable();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
        ViewCategories.class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        refreshBtn.setText(resourceMap.getString("refreshBtn.text")); // NOI18N
        refreshBtn.setName("refreshBtn"); // NOI18N
        refreshBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                refreshBtnActionPerformed(evt);
            }
        });

        exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
        exitBtn.setName("exitBtn"); // NOI18N
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitBtnActionPerformed(evt);
            }
        });

        categoryScrlPane.setName("categoryScrlPane"); // NOI18N
```

```

categoryTbl.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
        {null, null},
        {null, null}
    },
    new String [] {
        "Category ID", "Category Name"
    }
));
categoryTbl.setName("categoryTbl"); // NOI18N
categoryScrollPane.setViewportView(categoryTbl);
categoryTbl.getColumnModel().getColumn(0).setHeaderValue(resourceMap.getString(
    "categoryTbl.columnModel.title0")); // NOI18N
categoryTbl.getColumnModel().getColumn(1).setHeaderValue(resourceMap.getString(
    "categoryTbl.columnModel.title1")); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(refreshBtn)
                .addComponent(exitBtn))
            .addGap(10, 10, 10)
        )
);
jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addComponent(categoryScrollPane, javax.swing.GroupLayout.DEFAULT_SIZE, 248,
                Short.MAX_VALUE)
            .addGap(10, 10, 10)
            .addComponent(exitBtn)
            .addGap(10, 10, 10)
            .addComponent(refreshBtn)
            .addGap(10, 10, 10)
        )
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGap(10, 10, 10)
        )
);

```

```

    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
    );

    jPanell1.getAccessibleContext().setAccessibleName(resourceMap.getString(
        "jPanell1.AccessibleContext.accessibleName")); // NOI18N

    pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST:event_exitBtnActionPerformed
    dispose();
    // GEN-LAST:event_exitBtnActionPerformed

    /**
     * this method refreshes the table for when the categories have been added,
     * edited or removed elsewhere
     */
    private void refreshBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_refreshBtnActionPerformed
        // set select statement
        CG41App.dbObject.sqlString = "select COUNT(*) from category";
        thisError = CG41App.dbObject.getCountBySelect(); // execute count

        numRecords = CG41App.dbObject.NumberOfRecords; // save number of records

        // Construct an SQL statment to find all categories
        CG41App.dbObject.sqlString =
            "select CatID, CategoryName from category";
        // Execute the request
        thisError = CG41App.dbObject.getRecordSetBySelect();
        if (thisError == 0) {
            // No error on execution of statement
            // In order to make a table, you need to create an array of objects
            // It must be as long as the number of records you've counted and
            // must be as wide as the number of columns you've asked for
            Object[][] myTableData = new Object[numRecords][2];
            String[] myColumnNames = {"Category ID", "Category Name"};
            try {
                int row = 0;
                while (CG41App.dbObject.rs.next()) // For each record
                {
                    // get each part of the record from the record set
                    catID = CG41App.dbObject.rs.getString("CatID");
                    catName = CG41App.dbObject.rs.getString("CategoryName");
                    myTableData[row][0] = catID;
                    myTableData[row][1] = catName;
                }
            }
        }
    }
}

```

```

        row = row + 1;                // Next row in table
    }

    // Have the table data, so make a new table with it
    JTable jtblAllCustomers = new JTable(myTableData, myColumnNames);
    jtblAllCustomers.setFillViewportHeight(true);
    categoryScrlPane.getViewport().add(jtblAllCustomers);
    categoryScrlPane.repaint();
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    thisError = ex.getErrorCode();

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
}

} //GEN-LAST:event_refreshBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(ViewCategories.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(ViewCategories.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(ViewCategories.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(ViewCategories.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    }
}
//</editor-fold>

```

```
/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {

    public void run() {
        new ViewCategories().setVisible(true);
    }
});
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JScrollPane categoryScrlPane;
private javax.swing.JTable categoryTbl;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton refreshBtn;
// End of variables declaration//GEN-END:variables
}
```



```
package cg41;

import java.sql.SQLException;
import java.util.Date;
import javax.swing.JTable;

public class ViewCompletedCoursesByCourse extends javax.swing.JFrame {

    String myLocalString, govID, myCourseArray[], courseName, govName,
        course, myCourse;
    int thisError, numRecords, myCourseID, i = 0;
    Date dateTaken;

    public ViewCompletedCoursesByCourse () {
        initComponents();
        loadCourses();
    }

    /** This method is called from within the constructor to
     *  initialise the form.
     *  WARNING: Do NOT modify this code. The content of this method is
     *  always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents () {

        jPanel1 = new javax.swing.JPanel();
        refreshBtn = new javax.swing.JButton();
        exitBtn = new javax.swing.JButton();
        courseScrlPane = new javax.swing.JScrollPane();
        courseTbl = new javax.swing.JTable();
        compCourseCombo = new javax.swing.JComboBox();
        selectLbl = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
        ViewCompletedCoursesByCourse.class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        refreshBtn.setText(resourceMap.getString("refreshBtn.text")); // NOI18N
        refreshBtn.setName("refreshBtn"); // NOI18N
        refreshBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                refreshBtnActionPerformed(evt);
            }
        });

        exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
        exitBtn.setName("exitBtn"); // NOI18N
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
```

-2-

```

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(compCourseCombox, 0, 196, Short.MAX_VALUE)))
    .addContainerGap())
);

jPanell1Layout.setVerticalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanell1Layout.
        createSequentialGroup()
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(compCourseCombox, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                PREFERRED_SIZE)
            .addComponent(selectLbl))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 29,
            Short.MAX_VALUE)
        .addComponent(courseScrlPane, javax.swing.GroupLayout.PREFERRED_SIZE, 262,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(exitBtn)
            .addComponent(refreshBtn))
        .addContainerGap())
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
);

layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
);

jPanell1.getAccessibleContext().setAccessibleName(resourceMap.getString(
    "jPanell1.AccessibleContext.accessibleName")); // NOI18N

pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_exitBtnActionPerformed
    dispose();
}

```

```

} //GEN-LAST:event_exitBtnActionPerformed

/**
 * This method loads the courses into the combo box
 */
private void loadCourses() {
    CG41App.dbObject.sqlString = "select COUNT(*) from course"; //set query
    thisError = CG41App.dbObject.getCountBySelect(); //execute query
    numRecords = CG41App.dbObject.NumberOfRecords; //get number of records
    CG41App.dbObject.sqlString = "select CourseID, CourseName from course";
    thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
    if (thisError == 0) {
        try {
            for (i = 0; i < numRecords; i++) {
                //for amount of records, add each one to the combo box
                CG41App.dbObject.rs.next();
                course = CG41App.dbObject.rs.getString(1) + ", "
                    + CG41App.dbObject.rs.getString(2);
                compCourseCombox.addItem(course);
            }
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();

            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
    }
}

/**
 * this method refreshes the table for when courses have been added, edited
 * or removed elsewhere
 */
private void refreshBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_refreshBtnActionPerformed
    myCourse = compCourseCombox.getSelectedItem().toString(); //get selected course
    if (myCourse.equals("-")) {
        //show an error to the user
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("Please select a Course to search by.");
        MyMessage.setVisible(true);
    } else {
        myCourseArray = myCourse.split(","); //get the ID out
        myCourseID = Integer.parseInt(myCourseArray[0]); //store the ID

        //count the amount of records where the selected ID is concerned
        myLocalString = "select COUNT(*) from coursetaken where CourseID = "
            + myCourseID;
        CG41App.dbObject.sqlString = myLocalString;
        thisError = CG41App.dbObject.getCountBySelect(); //execute

        numRecords = CG41App.dbObject.NumberOfRecords; //save number of records
    }
}

```

```

// Construct an SQL statment to find all the information we need:
//The information that we're pulling here is the course name, datetaken,
//Governor ID and Governor First/Last names but only where the course ID
//selected and coursetaken course ID match.
//The joins allow us to pull the information from the three tables
myLocalString = "SELECT course.CourseName, coursetaken.DateTaken, "
    + "coursetaken.GovID, governor.Firstname, governor.Surname "
    + "FROM (coursetaken "
    + "INNER JOIN course ON course.CourseID = coursetaken.CourseID) "
    + "INNER JOIN governor ON governor.GovID = coursetaken.GovID "
    + "WHERE (coursetaken.CourseID = " + myCourseID + ")";
CG41App.dbObject.sqlString = myLocalString;

// Execute the query
thisError = CG41App.dbObject.getRecordSetBySelect();
if (thisError == 0) {
    // No error on execution of statement
    // In order to make a table, you need to create an array of objects
    // It must be as long as the number of records you've counted and
    // must be as wide as the number of columns you've asked for
    Object[][] myTableData = new Object[numRecords][4];
    String[] myColumnNames = {"Course Name", "Governor ID",
        "Governor Name", "Date Taken"};
    try {
        int row = 0;
        while (CG41App.dbObject.rs.next()) { // For each record
            //get each part of the record from the record set
            courseName = CG41App.dbObject.rs.getString("CourseName");
            govID = CG41App.dbObject.rs.getString("GovID");
            //concatonate the governor name
            govName = CG41App.dbObject.rs.getString("Firstname") + " "
                + CG41App.dbObject.rs.getString("Surname");
            dateTaken = CG41App.dbObject.rs.getDate("DateTaken");
            myTableData[row][0] = courseName;
            myTableData[row][1] = govID;
            myTableData[row][2] = govName;
            myTableData[row][3] = String.valueOf(dateTaken);
            row = row + 1;           // Next row in table
        }

        // Have the table data, so make a new table with it
        JTable jtblCoursesByID = new JTable(myTableData, myColumnNames);
        jtblCoursesByID.setFillsViewportHeight(true);
        courseScrlPane.getViewport().add(jtblCoursesByID);
        courseScrlPane.repaint();
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        thisError = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
}

```

```

    }
}
} //GEN-LAST:event_refreshBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByCourse.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByCourse.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByCourse.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByCourse.class.getName()).
        log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new ViewCompletedCoursesByCourse().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox compCourseCombox;
private javax.swing.JScrollPane courseScrlPane;
private javax.swing.JTable courseTbl;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton refreshBtn;
private javax.swing.JLabel selectLbl;
// End of variables declaration//GEN-END:variables
}

```

```
package cg41;

import java.sql.SQLException;
import java.util.Date;
import javax.swing.JTable;

public class ViewCompletedCoursesByGovernor extends javax.swing.JFrame {

    String myLocalString, courseID, myGovernorArray[], courseName, governorName,
        governor, myGovernor;
    int thisError, numRecords, myGovernorID, i = 0;
    Date dateTaken;

    public ViewCompletedCoursesByGovernor() {
        initComponents();
        loadGovernors();
    }

    /** This method is called from within the constructor to
     *  initialise the form.
     *  WARNING: Do NOT modify this code. The content of this method is
     *  always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        refreshBtn = new javax.swing.JButton();
        exitBtn = new javax.swing.JButton();
        courseScrlPane = new javax.swing.JScrollPane();
        courseTbl = new javax.swing.JTable();
        compGovCombox = new javax.swing.JComboBox();
        selectLbl = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(
        ViewCompletedCoursesByGovernor.class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        refreshBtn.setText(resourceMap.getString("refreshBtn.text")); // NOI18N
        refreshBtn.setName("refreshBtn"); // NOI18N
        refreshBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                refreshBtnActionPerformed(evt);
            }
        });

        exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
        exitBtn.setName("exitBtn"); // NOI18N
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
```

```
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            exitBtnActionPerformed(evt);
        }
    });

courseScrlPane.setName("courseScrlPane"); // NOI18N

courseTbl.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
        {null, null, null, null},
        {null, null, null, null}
    },
    new String [] {
        "Governor Name", "Course ID", "Course Name", "Date Taken"
    }
));
courseTbl.setName("courseTbl"); // NOI18N
courseScrlPane.setViewportViewView(courseTbl);
courseTbl.getColumnModel().getColumn(0).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title0")); // NOI18N
courseTbl.getColumnModel().getColumn(1).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title1")); // NOI18N
courseTbl.getColumnModel().getColumn(2).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title2")); // NOI18N
courseTbl.getColumnModel().getColumn(3).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title3")); // NOI18N

compGovCombox.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "-" }));
compGovCombox.setToolTipText(resourceMap.getString("compGovCombox.toolTipText")); //
NOI18N
compGovCombox.setName("compGovCombox"); // NOI18N

selectLbl.setText(resourceMap.getString("selectLbl.text")); // NOI18N
selectLbl.setName("selectLbl"); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                        .add(jPanel1Layout.createSequentialGroup()
                            .add(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .add(jPanel1Layout.createSequentialGroup()
                                    .addContainerGap()
                                    .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                        .add(jPanel1Layout.createSequentialGroup()
                                            .addComponent(courseScrlPane, javax.swing.GroupLayout.Alignment.LEADING)
                                            .addGroup(jPanel1Layout.createSequentialGroup()
                                                .addComponent(refreshBtn)
                                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                                                .addComponent(exitBtn))))
                                .add(jPanel1Layout.createSequentialGroup()
                                    .addGap(186, 186, 186)
                                    .addComponent(selectLbl)

```



```

        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addComponent(compGovCombox, 0, 185, Short.MAX_VALUE)))
    .addContainerGap())
);

jPanell1Layout.setVerticalGroup(
    jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanell1Layout.
        createSequentialGroup()
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(compGovCombox, javax.swing.GroupLayout.PREFERRED_SIZE,
                javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.
                PREFERRED_SIZE)
            .addComponent(selectLbl))
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 29,
            Short.MAX_VALUE)
        .addComponent(courseScrlPane, javax.swing.GroupLayout.PREFERRED_SIZE, 262,
            javax.swing.GroupLayout.PREFERRED_SIZE)
        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
        .addGroup(jPanell1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment
            .BASELINE)
            .addComponent(exitBtn)
            .addComponent(refreshBtn))
        .addContainerGap())
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
);

layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(layout.createSequentialGroup()
        .addComponent(jPanell1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
            GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
);

jPanell1.getAccessibleContext().setAccessibleName(resourceMap.getString(
    "jPanell1.AccessibleContext.accessibleName")); // NOI18N

pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_exitBtnActionPerformed
    dispose();
}

```

```

} //GEN-LAST:event_exitBtnActionPerformed

/**
 * This method loads the governors into the combo box
 */
private void loadGovernors() {
    CG41App.dbObject.sqlString = "select COUNT(*) from governor"; //get count
    thisError = CG41App.dbObject.getCountBySelect(); //execute count query
    numRecords = CG41App.dbObject.NumberOfRecords; //get number of records
    CG41App.dbObject.sqlString = "select GovID, Firstname, Surname from governor";
    thisError = CG41App.dbObject.getRecordSetBySelect(); //execute query
    if (thisError == 0) {
        try {
            for (i = 0; i < numRecords; i++) {
                //concatenate governor and add to the combo box for amount of records
                CG41App.dbObject.rs.next();
                governor = CG41App.dbObject.rs.getString(1) + ", " +
                    CG41App.dbObject.rs.getString(2) + " " +
                    CG41App.dbObject.rs.getString(3);
                compGovCombox.addItem(governor);
            }
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();

            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
    }

}

/**
 * this method refreshes the table for when courses have been added, edited
 * or removed elsewhere
 */
private void refreshBtnActionPerformed(java.awt.event.ActionEvent evt) {
    //GEN-FIRST:event_refreshBtnActionPerformed
    myGovernor = compGovCombox.getSelectedItem().toString(); //get selected course
    if (myGovernor.equals("-")) {
        //show an error to the user
        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("Please select a Governor to search by.");
        MyMessage.setVisible(true);
    } else {
        myGovernorArray = myGovernor.split(","); //get the ID out
        myGovernorID = Integer.parseInt(myGovernorArray[0]); //store the ID

        //count the amount of records where the selected ID is concerned
        myLocalString = "select COUNT(*) from coursetaken where GovID = " +
            myGovernorID;
        CG41App.dbObject.sqlString = myLocalString;
        thisError = CG41App.dbObject.getCountBySelect(); //execute
    }
}

```

```

numRecords = CG41App.dbObject.NumberOfRecords; //save number of records

// Construct an SQL statment to find all the information we need:
//The information that we're pulling here is the course name, datetaken,
//course ID, Governor ID and Governor First/Last names but only where the
//governor ID selected and coursetaken gocernor ID match.
//The joins allow us to pull the information from the three tables
myLocalString = "SELECT course.CourseName, coursetaken.DateTaken, "
    + "coursetaken.GovID, governor.Firstname, governor.Surname, "
    + "coursetaken.CourseID FROM (coursetaken "
    + "INNER JOIN course ON course.CourseID = coursetaken.CourseID) "
    + "INNER JOIN governor ON governor.GovID = coursetaken.GovID "
    + "WHERE (coursetaken.GovID = " + myGovernorID + ")";
CG41App.dbObject.sqlString = myLocalString;

// Execute the query
thisError = CG41App.dbObject.getRecordSetBySelect();
if (thisError == 0) {
    // No error on execution of statement
    // In order to make a table, you need to create an array of objects
    // It must be as long as the number of records you've counted and
    // must be as wide as the number of columns you've asked for
    Object[][] myTableData = new Object[numRecords][4];
    String[] myColumnNames = {"Governor Name", "Course ID",
        "Course Name", "Date Taken"};
    try {
        int row = 0;
        while (CG41App.dbObject.rs.next()) { // For each record
            //get each part of the record from the record set
            courseName = CG41App.dbObject.rs.getString("CourseName");
            courseID = CG41App.dbObject.rs.getString("CourseID");
            //concatonate the governor name
            governorName = CG41App.dbObject.rs.getString("Firstname") + " "
                + CG41App.dbObject.rs.getString("Surname");
            dateTaken = CG41App.dbObject.rs.getDate("DateTaken");
            myTableData[row][0] = governorName;
            myTableData[row][1] = courseID;
            myTableData[row][2] = courseName;
            myTableData[row][3] = String.valueOf(dateTaken);
            row = row + 1;           // Next row in table
        }

        // Have the table data, so make a new table with it
        JTable jtTblAllCustomers = new JTable(myTableData, myColumnNames);
        jtTblAllCustomers.setFillsViewportHeight(true);
        courseScrlPane.getViewport().add(jtTblAllCustomers);
        courseScrlPane.repaint();
    } catch (SQLException ex) {
        // Show errors in console
        System.out.println("SQLException: " + ex.getMessage());
        System.out.println("SQLState: " + ex.getSQLState());
        System.out.println("VendorError: " + ex.getErrorCode());
        thisError = ex.getErrorCode();

        UserMessageBox MyMessage = new UserMessageBox();
        MyMessage.setMessage("SQLException: " + ex.getMessage());
        MyMessage.setVisible(true);
    }
}

```

```

    }
}

}

} //GEN-LAST:event_refreshBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByGovernor.class.getName
        ()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByGovernor.class.getName
        ()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByGovernor.class.getName
        ()).log(java.util.logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(ViewCompletedCoursesByGovernor.class.getName
        ()).log(java.util.logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new ViewCompletedCoursesByGovernor().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JComboBox compGovCombox;
private javax.swing.JScrollPane courseScrlPane;
private javax.swing.JTable courseTbl;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton refreshBtn;
private javax.swing.JLabel selectLbl;
// End of variables declaration//GEN-END:variables

```

}

```
package cg41;

import java.sql.SQLException;
import javax.swing.JTable;

public class ViewCourses extends javax.swing.JFrame {

    String myLocalString, courseID, catID, courseName, catName;
    int thisError, numRecords = 0;

    public ViewCourses() {
        initComponents();
    }

    /** This method is called from within the constructor to
     *  initialise the form.
     *  WARNING: Do NOT modify this code. The content of this method is
     *  always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        refreshBtn = new javax.swing.JButton();
        exitBtn = new javax.swing.JButton();
        courseScrlPane = new javax.swing.JScrollPane();
        courseTbl = new javax.swing.JTable();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(ViewCourses.
        class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        refreshBtn.setText(resourceMap.getString("refreshBtn.text")); // NOI18N
        refreshBtn.setName("refreshBtn"); // NOI18N
        refreshBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                refreshBtnActionPerformed(evt);
            }
        });

        exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
        exitBtn.setName("exitBtn"); // NOI18N
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitBtnActionPerformed(evt);
            }
        });

        courseScrlPane.setName("courseScrlPane"); // NOI18N
```

```

courseTbl.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
        {null, null, null},
        {null, null, null}
    },
    new String [] {
        "Course ID", "Category Name", "Course Name"
    }
));
courseTbl.setName("courseTbl"); // NOI18N
courseScrollPane.setViewportView(courseTbl);
courseTbl.getColumnModel().getColumn(0).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title0")); // NOI18N
courseTbl.getColumnModel().getColumn(1).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title1")); // NOI18N
courseTbl.getColumnModel().getColumn(2).setHeaderValue(resourceMap.getString(
"courseTbl.columnModel.title2")); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap()
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(jPanel1Layout.createSequentialGroup()
                    .addComponent(refreshBtn)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                    .addComponent(exitBtn)
                    .addComponent(courseScrollPane))
                .addGap())
            .addContainerGap())
);
jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap()
            .addComponent(courseScrollPane, javax.swing.GroupLayout.DEFAULT_SIZE, 248,
Short.MAX_VALUE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(exitBtn)
                .addComponent(refreshBtn))
            .addContainerGap())
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap()
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.

```

```

        GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        .addContainerGap())
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanell, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addContainerGap())
        );

    jPanell.getAccessibleContext().setAccessibleName(resourceMap.getString(
        "jPanell.AccessibleContext.accessibleName")); // NOI18N

    pack();
} // </editor-fold> // GEN-END: initComponents

/**
 * This method closes the window
 */
private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST: event_exitBtnActionPerformed
    dispose();
    // GEN-LAST: event_exitBtnActionPerformed

    /**
     * This method will take a category ID as an input and will return
     * the category name linked to the category ID from the database
     */
    public String getCategoryName(int catID) {
        myLocalString = ""; // reset variables
        thisError = 0;
        // create search statement to find the name by the given ID
        myLocalString = "select * from category where CatID = '"
            + catID + "'";
        CG41App.dbObject2.sqlString = myLocalString; // set the statement
        thisError = CG41App.dbObject2.getRecordSetBySelect(); // execute
        try {
            CG41App.dbObject2.rs.next(); // get the name from the record set
            catName = CG41App.dbObject2.rs.getString("CategoryName");
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();
            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
        return catName;
    }

    /**
     * this method refreshes the table for when courses have been added, edited
     * or removed elsewhere

```



```

*/
private void refreshBtnActionPerformed(java.awt.event.ActionEvent evt) {
//GEN-FIRST:event_refreshBtnActionPerformed
    //count the amount of records
    CG41App.dbObject.sqlString = "select COUNT(*) from course";
    thisError = CG41App.dbObject.getCountBySelect();

    numRecords = CG41App.dbObject.NumberOfRecords; //save number of records

    // Construct an SQL statment to find all the categories
    CG41App.dbObject.sqlString =
        "select CourseID, CatID, CourseName from course";
    // Execute the query
    thisError = CG41App.dbObject.getRecordSetBySelect();
    if (thisError == 0) {
        // No error on execution of statement
        // In order to make a table, you need to create an array of objects
        // It must be as long as the number of records you've counted and
        // must be as wide as the number of columns you've asked for
        Object[][] myTableData = new Object[numRecords][3];
        String[] myColumnNames = {"Course ID", "Category ID", "Course Name"};
        try {
            int row = 0;
            while (CG41App.dbObject.rs.next()){ // For each record
                //get each part of the record from the record set
                courseID = CG41App.dbObject.rs.getString("CourseID");
                catID = CG41App.dbObject.rs.getString("CatID");
                //get the category name with the category ID
                catName = getCategoryName(Integer.parseInt(catID));
                courseName = CG41App.dbObject.rs.getString("CourseName");
                myTableData[row][0] = courseID;
                myTableData[row][1] = catName;
                myTableData[row][2] = courseName;
                row = row + 1;          // Next row in table
            }

            // Have the table data, so make a new table with it
            JTable jtblAllCustomers = new JTable(myTableData, myColumnNames);
            jtblAllCustomers.setFillsViewportHeight(true);
            courseScrlPane.getViewport().add(jtblAllCustomers);
            courseScrlPane.repaint();
        } catch (SQLException ex) {
            // Show errors in console
            System.out.println("SQLException: " + ex.getMessage());
            System.out.println("SQLState: " + ex.getSQLState());
            System.out.println("VendorError: " + ex.getErrorCode());
            thisError = ex.getErrorCode();

            UserMessageBox MyMessage = new UserMessageBox();
            MyMessage.setMessage("SQLException: " + ex.getMessage());
            MyMessage.setVisible(true);
        }
    }
} //GEN-LAST:event_refreshBtnActionPerformed

/**
 * @param args the command line arguments

```

```

*/
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(ViewCourses.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(ViewCourses.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(ViewCourses.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(ViewCourses.class.getName()).log(java.util.
        logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new ViewCourses().setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JScrollPane courseScrlPane;
private javax.swing.JTable courseTbl;
private javax.swing.JButton exitBtn;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton refreshBtn;
// End of variables declaration//GEN-END:variables
}

```

```
package cg41;

import java.sql.SQLException;
import java.util.Date;
import javax.swing.JTable;

public class ViewGovernors extends javax.swing.JFrame {

    String govID, firstname, surname;
    Date elecDate, endDate;
    int thisError, numRecords = 0;

    public ViewGovernors() {
        initComponents();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        refreshBtn = new javax.swing.JButton();
        exitBtn = new javax.swing.JButton();
        governorScrlPane = new javax.swing.JScrollPane();
        governorTbl = new javax.swing.JTable();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(ViewGovernors
        .class);
        setTitle(resourceMap.getString("Form.title")); // NOI18N
        setName("Form"); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(resourceMap.getString(
        "jPanel1.border.title"))); // NOI18N
        jPanel1.setName("jPanel1"); // NOI18N

        refreshBtn.setText(resourceMap.getString("refreshBtn.text")); // NOI18N
        refreshBtn.setName("refreshBtn"); // NOI18N
        refreshBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                refreshBtnActionPerformed(evt);
            }
        });

        exitBtn.setText(resourceMap.getString("exitBtn.text")); // NOI18N
        exitBtn.setName("exitBtn"); // NOI18N
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitBtnActionPerformed(evt);
            }
        });
    }
}
```

```

governorScrlPane.setName("governorScrlPane"); // NOI18N

governorTbl.setModel(new javax.swing.table.DefaultTableModel(
    new Object [][] {
        {null, null, null, null, null},
        {null, null, null, null, null}
    },
    new String [] {
        "Governor ID", "First Name", "Last Name", "Elected Date", "End Date"
    }
));
governorTbl.setName("governorTbl"); // NOI18N
governorScrlPane.setViewportViewView(governorTbl);
governorTbl.getColumnModel().getColumn(0).setHeaderValue(resourceMap.getString(
    "governorTbl.columnModel.title0")); // NOI18N
governorTbl.getColumnModel().getColumn(1).setHeaderValue(resourceMap.getString(
    "governorTbl.columnModel.title1")); // NOI18N
governorTbl.getColumnModel().getColumn(2).setHeaderValue(resourceMap.getString(
    "governorTbl.columnModel.title2")); // NOI18N
governorTbl.getColumnModel().getColumn(3).setHeaderValue(resourceMap.getString(
    "governorTbl.columnModel.title3")); // NOI18N
governorTbl.getColumnModel().getColumn(4).setHeaderValue(resourceMap.getString(
    "governorTbl.columnModel.title4")); // NOI18N

javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(refreshBtn)
                .addComponent(exitBtn)
                .addComponent(governorScrlPane))
            .addContainerGap(10, true))
);
jPanel1Layout.setVerticalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel1Layout.createSequentialGroup()
            .addGap(10, 10, 10)
            .addComponent(governorScrlPane, javax.swing.GroupLayout.DEFAULT_SIZE, 248,
                Short.MAX_VALUE)
            .addGap(10, 10, 10)
            .addComponent(exitBtn)
            .addComponent(refreshBtn)
            .addContainerGap(10, true))
);

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());

```

```

        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(10, 10, 10)
                    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                        GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGap(10, 10, 10))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(10, 10, 10)
                    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.
                        GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGap(10, 10, 10))
        );

        jPanel1.getAccessibleContext().setAccessibleName(resourceMap.getString(
            "jPanel1.AccessibleContext.accessibleName")); // NOI18N

        pack();
    } // </editor-fold> // GEN-END: initComponents

    /**
     * This method is used to close the window when the exit button is pressed
     */
    private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST: event_exitBtnActionPerformed
        dispose();
    } // GEN-LAST: event_exitBtnActionPerformed

    /**
     * this method refreshes the table for when governors have been added, edited
     * or removed elsewhere
     */
    private void refreshBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST: event_refreshBtnActionPerformed
        // set select statement
        CG41App.dbObject.sqlString = "select COUNT(*) from governor";
        thisError = CG41App.dbObject.getCountBySelect(); // execute count by select

        numRecords = CG41App.dbObject.NumberOfRecords; // save number of records

        // Construct an SQL statment to find all governors
        CG41App.dbObject.sqlString =
            "select GovID, Firstname, "
            + "Surname, ElectionDate, EndDate from governor";
        // Execute the query
        thisError = CG41App.dbObject.getRecordSetBySelect();
        if (thisError == 0) {
            // No error on execution of statement
            // In order to make a table, you need to create an array of objects
            // It must be as long as the number of records you've counted and
            // must be as wide as the number of columns you've asked for
            Object[][] myTableData = new Object[numRecords][5];
            String[] myColumnNames = {"Governor ID", "First Name", "Surname",
                "Elected Date", "End Date"};

```

```

try {
    int row = 0;
    while (CG41App.dbObject.rs.next()) // For each record
    {
        //get each part of the governor from the Record Set
        govID = CG41App.dbObject.rs.getString("GovID");
        firstname = CG41App.dbObject.rs.getString("Firstname");
        surname = CG41App.dbObject.rs.getString("Surname");
        elecDate = CG41App.dbObject.rs.getDate("ElectionDate");
        endDate = CG41App.dbObject.rs.getDate("EndDate");
        myTableData[row][0] = govID;
        myTableData[row][1] = firstname;
        myTableData[row][2] = surname;
        myTableData[row][3] = String.valueOf(elecDate);
        myTableData[row][4] = String.valueOf(endDate);
        row = row + 1;           // Next row in table
    }

    // Have the table data, so make a new table with it
    JTable jtblAllGovernors = new JTable(myTableData, myColumnNames);
    jtblAllGovernors.setFillsViewportHeight(true);
    governorScrlPane.getViewport().add(jtblAllGovernors);
    governorScrlPane.repaint();
} catch (SQLException ex) {
    // Show errors in console
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
    thisError = ex.getErrorCode();

    UserMessageBox MyMessage = new UserMessageBox();
    MyMessage.setMessage("SQLException: " + ex.getMessage());
    MyMessage.setVisible(true);
}
}

} //GEN-LAST:event_refreshBtnActionPerformed

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional)
    ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look
    and feel.
    * For details see
    http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
    */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.
        getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    }
}

```

```
    } catch (ClassNotFoundException ex) {
        java.util.logging.Logger.getLogger(ViewGovernors.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (InstantiationException ex) {
        java.util.logging.Logger.getLogger(ViewGovernors.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (IllegalAccessException ex) {
        java.util.logging.Logger.getLogger(ViewGovernors.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    } catch (javax.swing.UnsupportedLookAndFeelException ex) {
        java.util.logging.Logger.getLogger(ViewGovernors.class.getName()).log(java.util.
            logging.Level.SEVERE, null, ex);
    }
    //</editor-fold>

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new ViewGovernors().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JButton exitBtn;
private javax.swing.JScrollPane governorScrlPane;
private javax.swing.JTable governorTbl;
private javax.swing.JPanel jPanel1;
private javax.swing.JButton refreshBtn;
// End of variables declaration//GEN-END:variables
}
```

```
package cg41;

public class Validation {

    private boolean isInDatabase;

    public Validation() {
    }

    /**
     * This method requires a string input for the first name and will return a
     * boolean value as to whether or not it matches the validation pattern
     */
    public boolean validateFirstName(String inFirstName) {
        return inFirstName.matches("[A-Z][a-zA-Z]{1,9}");
    }

    /**
     * This method requires a string input for the last name and will return a
     * boolean value as to whether or not it matches the validation pattern
     */
    public boolean validateSurname(String inSurnameName) {
        return inSurnameName.matches("([a-zA-z' -]{1,2})([a-zA-Z ' -]{1,18})");
    }

    /**
     * This method requires a string input for the course name and will return a
     * boolean value as to whether or not it matches the validation pattern
     */
    public boolean validateCourseName(String inCourseName) {
        return inCourseName.matches("([a-zA-z]{1,2})([a-zA-Z0-9 ' -]{1,20})");
    }

    /**
     * This method requires a string input for the category name and will return a
     * boolean value as to whether or not it matches the validation pattern
     */
    public boolean validateCategoryName(String inCatName) {
        return inCatName.matches("([a-zA-z]{1,2})([a-zA-Z0-9 ' -]{1,20})");
    }

    public boolean isGovInDB(String inName) {
        isInDatabase = false;
        return isInDatabase;
    }

    public boolean isCourseInDB(String inName) {
        isInDatabase = false;
        return isInDatabase;
    }

    public boolean isCatInDB(String inName) {
        isInDatabase = false;
        return isInDatabase;
    }
}
```



```
/*
 * CG41AboutBox.java
 */

package cg41;

import org.jdesktop.application.Action;

public class CG41AboutBox extends javax.swing.JDialog {

    public CG41AboutBox(java.awt.Frame parent) {
        super(parent);
        initComponents();
        getRootPane().setDefaultButton(closeButton);
    }

    @Action public void closeAboutBox() {
        dispose();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN: initComponents
    private void initComponents() {

        closeButton = new javax.swing.JButton();
        javax.swing.JLabel appTitleLabel = new javax.swing.JLabel();
        javax.swing.JLabel versionLabel = new javax.swing.JLabel();
        javax.swing.JLabel appVersionLabel = new javax.swing.JLabel();
        javax.swing.JLabel vendorLabel = new javax.swing.JLabel();
        javax.swing.JLabel appVendorLabel = new javax.swing.JLabel();
        javax.swing.JLabel appDescLabel = new javax.swing.JLabel();
        javax.swing.JLabel imageLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        org.jdesktop.application.ResourceMap resourceMap = org.jdesktop.application.
        Application.getInstance(cg41.CG41App.class).getContext().getResourceMap(CG41AboutBox.
        class);
        setTitle(resourceMap.getString("title")); // NOI18N
        setModal(true);
        setName("aboutBox"); // NOI18N
        setResizable(false);

        javax.swing.ActionMap actionMap = org.jdesktop.application.Application.getInstance(
        cg41.CG41App.class).getContext().getActionMap(CG41AboutBox.class, this);
        closeButton.setAction(actionMap.get("closeAboutBox")); // NOI18N
        closeButton.setName("closeButton"); // NOI18N

        appTitleLabel.setFont(appTitleLabel.getFont().deriveFont(appTitleLabel.getFont().
        getStyle() | java.awt.Font.BOLD, appTitleLabel.getFont().getSize()+4));
        appTitleLabel.setText(resourceMap.getString("Application.title")); // NOI18N
        appTitleLabel.setName("appTitleLabel"); // NOI18N

        versionLabel.setFont(versionLabel.getFont().deriveFont(versionLabel.getFont().
```

```

getStyle() | java.awt.Font.BOLD));
versionLabel.setText(resourceMap.getString("versionLabel.text")); // NOI18N
versionLabel.setName("versionLabel"); // NOI18N

appVersionLabel.setText(resourceMap.getString("Application.version")); // NOI18N
appVersionLabel.setName("appVersionLabel"); // NOI18N

vendorLabel.setFont(vendorLabel.getFont().deriveFont(vendorLabel.getFont().getStyle()
    | java.awt.Font.BOLD));
vendorLabel.setText(resourceMap.getString("vendorLabel.text")); // NOI18N
vendorLabel.setName("vendorLabel"); // NOI18N

appVendorLabel.setText(resourceMap.getString("Application.vendor")); // NOI18N
appVendorLabel.setName("appVendorLabel"); // NOI18N

appDescLabel.setText(resourceMap.getString("appDescLabel.text")); // NOI18N
appDescLabel.setName("appDescLabel"); // NOI18N

imageLabel.setIcon(resourceMap.getIcon("imageLabel.icon")); // NOI18N
imageLabel.setName("imageLabel"); // NOI18N

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addComponent(imageLabel)
            .addGap(18, 18, 18)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                .addGroup(javax.swing.GroupLayout.Alignment.LEADING, layout.
                    createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.
                            Alignment.LEADING)
                            .addComponent(versionLabel)
                            .addComponent(vendorLabel))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.
                            Alignment.LEADING)
                            .addComponent(appVersionLabel)
                            .addComponent(appVendorLabel))
                        .addGap(102, 102, 102))
                .addComponent(appTitleLabel, javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(appDescLabel, javax.swing.GroupLayout.Alignment.LEADING,
                    javax.swing.GroupLayout.DEFAULT_SIZE, 270, Short.MAX_VALUE)
                .addComponent(closeButton))
            .addContainerGap())
        );
layout.setVerticalGroup(
    layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addComponent(imageLabel, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.
                GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(appTitleLabel)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(appDescLabel, javax.swing.GroupLayout.PREFERRED_SIZE, javax.

```

```
swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
BASELINE)
    .addComponent(versionLabel)
    .addComponent(appVersionLabel))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.
BASELINE)
    .addComponent(vendorLabel)
    .addComponent(appVendorLabel))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 53,
Short.MAX_VALUE)
.addComponent(closeButton)
.addContainerGap()

);

pack();
} // </editor-fold> // GEN-END: initComponents

// Variables declaration - do not modify // GEN-BEGIN: variables
private javax.swing.JButton closeButton;
// End of variables declaration // GEN-END: variables
}
```