# **CMPUT 391**

# **Project: Ocean Observation System**

# **Group 18:** Huy Truong, Wayne Choi, Akram Hassen

**General System Architecture**

project folder

|---WEB-INF

| |---classes

| |---util

| | |---\*object.java

| |---\*servlet.java

|---\*.jsp

All modules: Login, Sensor and user management, Subscribe, Uploading, Search, Data analysis.

Each module is constructed by its .jsp files and .java files.

The .jsp files are front-end of the system that display the user interface as a website to receive actions from users then send to the back-end that are handled by servlet classes in .java files. Most of the sql are stored in Db.java in the util folder.

One special note is that sensor.jsp can be accessed by all users, but depending on what role the user is, it will change the functionality to suite the user.

### **Login Module**

### The login module is contains two parts, the login section and personal/password change section. In the login section the user is given inputs to input a username and password. Depending on what type

### the user is he or she is given certain areas of the website to view.

The login module contains: login.jsp, LoginServlet.java, change\_info.jsp, UpdatePassServlet.java, UpdateInfoServlet.java

login.jsp will display a form for user to enter his user\_name and password. Once entered the user is able to traverse the site, whereas if the user tried to enter any other site they would have been kicked back to the login page.

LoginServlet.java to handles the back-end for login.jsp where all the required queries are accessed through Db.java. This will grab the username from the User's table and compares the password entered with what is stores in the database.

change\_info.jsp is accessed by everyone regardless of the type of role the user has. This front-end is where depending on how the page was accessed there will be different views. This page can be accessed by clicking Account Settings. One view to change personal information and the other to change username and password. This may look like two different pages, but it will change dynamically based on the selection made by the user.

UpdatePassServlet.java and UpdateInfoServlet.java are the back-end for change\_info.jsp. UpdatePassServlet.java changes username/password and UpdateInfoServlet.java changes personal information.

SQL USED:

Match password to user: "SELECT password FROM users where user\_name = '" + username + "'"

Change Password: "UPDATE users SET password = '"+password+"' WHERE user\_name = '" + username + "'"

Change User: "UPDATE users SET user\_name = '" + n\_user\_name + "' WHERE user\_name = '" + user\_name + " '"

Change First Name: "UPDATE persons SET first\_name = '" +fname + "' WHERE person\_id= '" + person\_id + " '"

Change Last Name: "UPDATE persons SET last\_name = '" +lname+ "' WHERE person\_id = '" +person\_id +" '"

Change Address: "UPDATE persons SET address = '" +address+ "' WHERE person\_id = '" + person\_id+ "'"

Change Phone Number: "UPDATE persons SET phone = '" + phone + "' WHERE person\_id = '" + person\_id + "'"

### **Search Module**

### This module allows the scientist and only the scientist to search from the database for sensors that they are subscribed to between two different dates.

### The Search module contains: search.jsp and SearchServlet.java

### Search, jsp is front end to the module where the scientist can can search for different sensor data that they are subscribed to. There are four fields, keywords/value, sensor type, location, and from and to dates.

### SearchServlet.java contains all the methods that grabs all the search functions. All of the search SQL functions are stored in the db in a file called db.java.

### SQL USED:

Search Audio: “SELECT au.\* FROM audio\_recordings au JOIN subscriptions su on au.sensor\_id = su.sensor\_id JOIN sensors s on au.sensor\_id = s.sensor\_id WHERE su.person\_id =” + person\_id + "AND au.description LIKE ' %"+keywords+" ' AND s.location LIKE ' %"+location+" ' AND au.date\_created BETWEEN TO\_DATE('"+fromdate+"', 'YYYY-MM-DD') AND TO\_DATE('"+todate+"', 'YYYY-MM-DD') ORDER BY au.recording\_id";

Search Image: "SELECT i.\* FROM images i JOIN subscriptions su on i.sensor\_id = su.sensor\_id JOIN sensors s on i.sensor\_id = s.sensor\_id WHERE su.person\_id =" + person\_id+"AND i.description LIKE '%"+keywords+" ' "+"AND s.location LIKE '%"+location+" ' "+ "AND i.date\_created BETWEEN TO\_DATE('"+fromdate+"', 'YYYY-MM-DD') " + "AND TO\_DATE('"+todate+"', 'YYYY-MM-DD') ORDER BY i.image\_id";

Search Scalar: "SELECT s.id,s.sensor\_id,s.date\_created,s.value FROM scalar\_data s JOIN subscriptions su on s.sensor\_id = su.sensor\_id JOIN sensors se on s.sensor\_id = se.sensor\_id WHERE su.person\_id =" + person\_id + "AND se.location LIKE ' %"+location+" ' " + "AND date\_created BETWEEN TO\_DATE('"+fromdate+"', 'YYYY-MM-DD') AND TO\_DATE('"+todate+"', 'YYYY-MM-DD') ORDER BY s.id";

This is searching when keywords search field is empty. If it is not add AND s.value =" + Float.parseFloat(keywords) or if keyword is not a number add AND s.value = null.

When searching without a any fields all three sqls above are merged using union.

### **Sensor and User Management Module**

This module allows only the admin to add/delete sensors and also the only one that can mange users. Managing the users mean creating, delete, or update user info. Even though any user can change their own information, they can also ask the admin to do it for them. Clicking the User and Sensor Management link in the header will bring up the different options to add/delete sensors and also the mange user accounts.

The Sensor and User Management Module contains: sensor.jsp, SensorServlet.java, register.jsp, RegServlet.java

sensor.jsp is the front-end to SensorServlet.java and it will change to fit the need of the administrator.

SensorServlet.java is the back-end and it will get all the required methods needed for adding and deleting sensors.

register.jsp contains two parts. One part is registering a new username for an existing person and creating a new person.

RegServlet.java is the back-end to register.jsp where all the methods needed to create users and new persons are created here.

SQL USED:

Inserting New Sensor: "INSERT INTO sensors Values(" + sensor\_id + ",'" + location +" ','" + sensor\_type + "','" + description + "')"

Inserting New Person: "INSERT INTO persons Values('" + personID + "','" + fname + "','" + lname + "','" + address + "','" + email + "','" + phone + "')"

Inserting New User: "INSERT INTO users Values('" + uname + "','" + pass + "','" + role + "','" + personID + "', CURRENT\_TIMESTAMP)";

### **Subscribe Module**

This module available only to the scientist allows the scientist to subscribe and unsubscribe to different sensors. Clicking the subscribe link on the header brings the user to the module.

The Subscribe Module contains: subscription.jsp, SubscriptionServlet.java

subscription.jsp is the front end of the subscription module where all the web interface will be. Here the user is given two options. One option is to add subscription and the other to delete subscription. Both will open pop-up boxes that will list sensors. The add subscription pop-up gives scientists a list of sensors that the has not been subscribed to. Adding one will remove it from the list. Delete subscription show the opposite and does the same thing as adding except subscribing to sensors

SubscriptionServlet.java, the back-end that will do all the needed work to get the subscription front-end to work. The java page contains method that accesses Db.java, which contains all the needed SQL statements.

SQL USED:

Delete subscription: "DELETE FROM subscriptions WHERE sensor\_id = " + sensor\_id+" AND person\_id = "+ person\_id

New Subscription: "INSERT INTO subscriptions Values("+sensor\_id + "," + person\_id + ")"

### **Uploading Module**

This module is only for the data manager. He or she will have the ability to add different sensor data types. The types are audio, image, and scalar data.

The upload module contains : sensor.jsp, SensorServlet,java, , ImageSevlet.java, ScalarServlet.java

Sensor.jsp is the main view for managing sensors the. For this module sensor.jsp changes it page to reflect that the user is a data manager. The left side will have a list links that can be clicked to add audio, image, or scalar CSV text.

SensorServlet.java is the back-end for Sensor.jsp

AudioServelet.java will output the audio file for the correct audio sensor.

ImageSevlet.java will output the image file for the correct image sensor..

ScalarServlet.java will output the correct CSV file that is needed.

SQL USED:

Adding audio: "INSERT INTO audio\_recordings VALUES("+recording\_id+"," +sensor\_id+",?,"+length+ ",'"+description+"', ?)"

Adding image: "INSERT INTO images VALUES("+image\_id+"," + sensor\_id +",?,'"+description+"', ?, ?)"

Adding Scalar: "INSERT INTO scalar\_data VALUES(" + id + "," + sensor\_id +", TO\_DATE('"+dateTimeLocal+"', 'DD/MM/YYYY HH24:MI:SS'),"+ value +")"