
Radiology Information System

Project Report

CMPUT 391 Winter 2015

Group:
Jack Chen
Brandon Williams
Christopher Li Sheung Ying

Login Module

login.html:

- Works with login.jsp.
- Displays to the user a login page which they can enter in their username and password to acquire access to the Radiology Information System.

login.jsp:

- It receives the username and password from login.html and redirects the logged in user to portal.jsp.
- It connects to the specified database and uses the SQL statement (1lo) to find the users with the received username and returns the data of each username found.
- It then parses the data connected to each username and checks to see if the password received from login.html matches anything. If it does not match anything an error message will be displayed and the user will have to try to login again.
- If the user successfully logs in they will be redirected to a portal page with links to other functions that they can use in the R.I.S.
- These functions will vary depending on what kind of user class they are (administrator, patient, doctor, and radiologist).
- Creates a session attributes of the user's record id, username, and class type(administrator, patient, doctor, radiologist).
- These session attributes will be used throughout the other jsp files within the login module.

portal.jsp:

- Is used after the user has successfully logged into the system.
- It retrieves the session attribute from login.jsp and it determines what links of functions to display.
- It also contains a link to edit the users personal information.
- The portal is also used when the user wishes to logout. At the top right corner of the each page there will be a link to the portal. Once they click on it there will be a link on the portal to logout.

edit_account.jsp:

- It allows the logged in user to edit his/her account information such as First Name, Last Name, Address, Email, Phone number, or Password change.
- It connects to the specified database and determines which value(s) need to be updated for the given user (found by getting the record id from the session attribute made in the login.jsp).
- It will detect which fields need to be updated and use the SQL statements (2lo, 3lo, 4lo, 5lo, 6lo, 7lo) and update the values in the database.
- Once the update is complete the user will be notified and the SQL statement (8lo) is used in the process to display the updated information to the user.

SQL Statements Used:

1lo. "SELECT * FROM users WHERE user_name = '"+user_name+"'";
2lo. "UPDATE persons SET first_name = '"+new_fn+"' WHERE person_id = "+id;
3lo. "UPDATE persons SET last_name = '"+new_ln+"' WHERE person_id = "+id;
4lo. "UPDATE persons SET address = '"+new_addr+"' WHERE person_id = "+id;
5lo. "UPDATE persons SET email = '"+new_email+"' WHERE person_id = "+id;
6lo. "UPDATE persons SET phone = '"+new_phone+"' WHERE person_id = "+id;
7lo. "UPDATE users SET password = '"+new_pass+"' WHERE person_id = "+id;
8lo. "SELECT * FROM persons WHERE person_id = "+id;

User Management Module

manage_users.jsp:

- Allows the administrator manage (to enter or update) the user information such as the information stored in tables *users,persons, family_doctor*.

- The admin could enter the person's id number so that they can manage their information (First Name, Last Name, Address, Email, or Phone).
- We connect to the specified database and use the SQL statement (1um) to find the specified person in the database if they exist.
- Once the person is found, it displays the person's ID, First Name, Last Name, Address, Email, and Phone. Next a form is displays below which allows the admin to update any of the fields.
- It will detect which fields need to be updated and use the SQL statements (2um, 3um, 4um, 5um, 6um) and update the values in the database for the specified target_id (the record_id).
- Once the update is complete the person will be notified and the SQL statement (7um) is used in the process to display the updated information to the admin.

- The admin could also enter the user's username to edit their user_name, password, or class.
- We connect to the specified database and use the SQL statement (8um) to find the specified user in the database if they exist.
- Once the user is found, it displays the user's Username, Password, Class, and PID. Next a form is displays below which allows the admin to update any of the fields.
- It will detect which fields need to be updated and use the SQL statements (9um, 10um, 11um) and update the values in the database for the specified target_un (the user_name).
- Once the update is complete the user will be notified and the SQL statement (12um) is used in the process to display the updated information to the admin.

SQL Statements Used:

1um. "SELECT * FROM persons WHERE person_id = "+request.getParameter("f_pid");
2um. "UPDATE persons SET first_name = '"+new_fn+"' WHERE person_id = "+target_id;
3um. "UPDATE persons SET last_name = '"+new_ln+"' WHERE person_id = "+target_id;
4um. "UPDATE persons SET address = '"+new_addr+"' WHERE person_id = "+target_id;
5um. "UPDATE persons SET email = '"+new_email+"' WHERE person_id = "+target_id;
6um. "UPDATE persons SET phone = '"+new_phone+"' WHERE person_id = "+target_id;
7um. "SELECT * FROM persons WHERE person_id = "+target_id;
8um. "SELECT * FROM users WHERE user_name = '"+request.getParameter("f_un")+"'";
9um. "UPDATE users SET user_name = '"+new_un+"' WHERE user_name = '"+target_un+"'";
10um. "UPDATE users SET password = '"+new_pass+"' WHERE user_name = '"+target_un+"'";
11um. "UPDATE users SET class = '"+new_class+"' WHERE user_name = '"+target_un+"'";
12um. "SELECT * FROM users WHERE user_name = '"+target_un+"'";

Report Generating Module

reportPrompt.jsp:

- Works with reportResult.jsp.
- Displays to the admin (no other class) a form where he/she can search for a specific diagnosis with a start date and end date. By entering into the fields the diagnosis, start date and end date.
- When the admin presses the submit button he/she is redirected to reportResult.jsp.

reportResult.jsp:

- Retrieves the input of diagnosis, start date and end date from reportPrompt.jsp.
- It then connects to the specified database and uses the SQL statement (1rg) to retrieve the rows which match the requirements entered by the admin.
- The values returned from the SQL statement are then displayed in a table for the admin.

SQL Statements Used:

1rg. "SELECT first_name, last_name, address, phone, test_date FROM persons p, radiology_record r WHERE p.person_id = r.patient_id AND r.diagnosis = '"+ diagnosis + "' AND r.test_date BETWEEN to_date('"+ start +"', 'DD/MM/YYYY') AND to_date('"+ end +"', 'DD/MM/YYYY') ORDER BY last_name";

Uploading Module

UploadImage.jsp:

- Works with UploadImage.java.
- Displays to the specific user (radiologist) a form where they can enter a Radiology Record ID and browse for an image that they want to upload that is on their computer. If the radiologist does not enter a valid Record ID along with a image file, they will receive an error message displayed to them.

UploadImage.java:

- Handles the image and the Record ID from UploadImage.jsp where it determines what to do with the information (Record ID and image file).
  Classes/Methods:
  - protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
    ⇒doPost is called when UploadImage.jsp is redirected to UploadImage.java.
    ⇒it uses the shrink() function to created different sizes (thumbnail, regular_size, full_size) of the image given for be put into the database (pacs_images).
    ⇒Uses the SQL statement (1up) to retrieve the number of Record IDs that match the Radiologist inputed one. If it cannot find a Record ID then it is not in the database.
    ⇒Uses the SQL statement (2up) to place the empty blobs in the database and then uses (3up) to write the images to the blobs.

  - public static BufferedImage shrink(BufferedImage image, int n)
    ⇒Shrinks the image by a given factor of n and returns the shrunk image.

  - private static Connection getConnected(String drivername, String dbstring,String username, String password) throws Exception
    ⇒Used to get a connect to the specified database.

SQL Statements Used:

1up. "select count(*) from radiology_record where record_id = " + record_id
2up. "INSERT INTO pacs_images VALUES(" + record_id + "," + image_id + ", empty_blob(), empty_blob(), empty_blob())"
3up. "SELECT * FROM pacs_images WHERE image_id = " + image_id + " FOR UPDATE"

Search Module

search.jsp:

- It allows the user to enter in a multiple sorting choices (time period, sorting test date by most recent or oldest) as well as a keyword to search for.
- The users choices are then saved and used by the rest of the program to created unique sqls that will display (in a table) the search information that the user was looking for.
- We use SQL statement (1sm) to retrieve the data that the user has requested (based on their input choices from the begin of the search).
- The SQL statement (1sm) can be dynamically changed depending on the input from the user.
- Once the SQL statement has been prepared we connect to the specific database, runs the SQL statement.
- If successful it will display to the user a table of record_ids, patient_ids, doctor_ids, test_type, prescribing_date, test_date, diagnosis, description and rank.
- Get the session which had the user class and person id.
- We verify it with the information from the record row to give the data the is specific to the class the user is.
- If a doctor logs in and searches for a record we will assume that the doctor_id from radiology record is the doctor of the patient of the record and so the search simply matches the logged in doctors id with the doctor id of the row.

SQL Statements Used:

1sm. SELECT DISTINCT 6*score(1) + 6*score(2) + 3*score(3) + score(4) as rank, r.record_id, r.patient_id, r.doctor_id, r.radiologist_id, r.test_type, r.prescribing_date, r.test_date, r.diagnosis, r.description FROM radiology_record r, persons p WHERE r.patient_id = p.person_id AND contains(p.first_name, ?, 1) > 0 OR contains(p.last_name, ?, 2) > 0 OR contains(r.diagnosis, ?, 3) > 0 OR contains(r.description, ?, 4) > 0 ORDER BY rank DESC;

Data Analysis Module

olapPrompt.jsp:

- Works with olapResults.jsp.
- Used by the admin only. Displays to the user three choices, Patient Name, Test Type, or Time Period that the user can choose any combination of the three via checkboxes. The selected choices will be stored as attributes so that sessions can get them when needed.
- If the user decides to use the Time Period option, the user will be able to select either week, month, or year for performing generalization (roll up) and specialization (drill down).
- When the user presses submit, they will be redirected to olapResult.jsp.

- Error messages will be displayed if the user does not select any options or if the week, month, or year is not selected with the user chooses to use Time Period.

olapResult.jsp:

- This will display the data found based on the choices chosen (Patient Name, Test Type, or Time Period) in a table format.
- It takes the values chosen from olapPrompt.jsp through session attributes and are processed through different conditions in olapResult.jsp.
- Based on the options selected by the user, different SQL statements are added dynamically to the main SQL string to create the statement (1da - 10da combination).
- It connects to the specified database and runs the main SQL statement and creates the table and displays it to the admin.

SQL Statements Used:

1da. "SELECT";
2da. "p.patient_id,";
3da. "r.test_type,";
4da. " TRUNC(r.test_date,'IW')";
5da. " TRUNC(r.test_date,'MM')";
6da. " TRUNC(r.test_date,'Y')";
7da. " COUNT(i.record_id) FROM persons p, radiology_record r, pacs_images i WHERE p.person_id = r.patient_id AND r.record_id = i.record_id" + group;
8da. p.person_id
9da. r.test_type
10da. test_date

(we can combine [1da] and [2da - 3da] and [one of 4da - 6da] and [7da] and [one of 8da - 10da] combinations for the main SQL statement)