Advanced Online Task System - Project Report

By
Praneeth Pulusani (prp9769@rit.edu)
Arjun Sohaney (axs4043@rit.edu)
Keyur Chitnis (kxc9778@rit.edu)

Online task system is a web application which let the users manage their tasks efficiently. The application is developed on JSP/J2EE platform. We have designed two xml schema. One is a template to keep track of new users and the other one is a template to maintain a list of tasks of a particular user. A new task file based on the schema is created for each user up on registration.

System Architecture:

High Level System Architecture
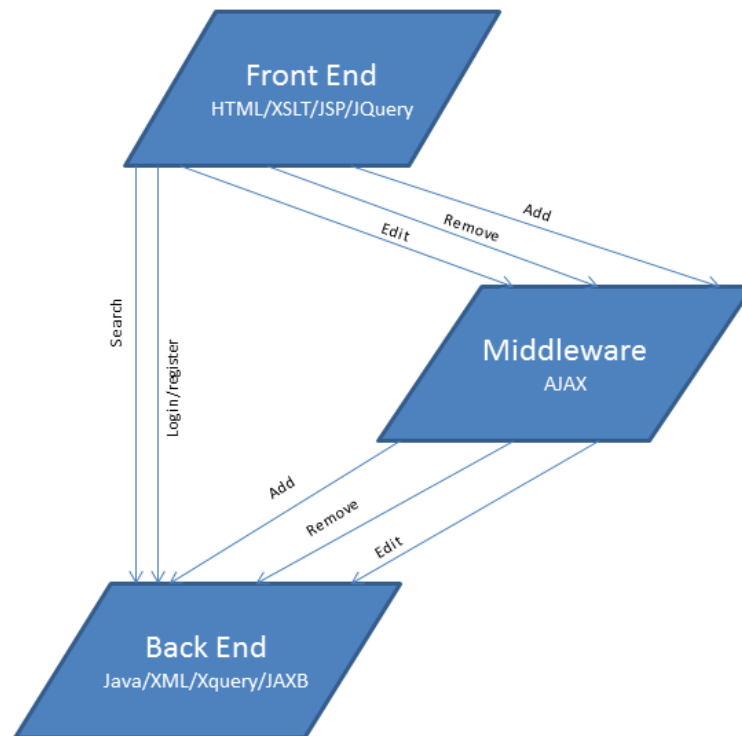
Features:

    1. Login: A user will be able to login to the task system using a unique username and password. JSP looks for a match of the input parameters in users.xml file. If there is a match, JSP loads the user's tasks data file into context and forwards the user into the

application. Otherwise, the user is taken back to the login page.

1a. Registration: A user will be able to register an account on the task system. When the user registers, an entry specified by the XML Schema is added to the users.xml file.

2. Adding / Editing/ Removing a task: The user can add tasks by entering details in the row with empty fields and clicking on the add button associated with that row. The user will also be able to remove a task by clicking the delete button. The user will be able to edit the task by clicking the edit button. When any of these buttons are clicked, AJAX is used to silently talk to the back-end java functions to process the action and re-render a part of the page.

3. Changing the status of a task: User will be able to mark the status of a task i.e. either complete or incomplete.

4. Tasks with cross-dependency: User can add task dependency to another task.
If Task G is dependent on Task A, if user tries to change the status of task G to completed, the
applications warns the user that Task A must be completed before Task G can be mark completed.

5. Due Date and priority field: User can assign due date and priority to all the tasks. Priority can have the following values Low, Medium, High.

6. RIT Themed: The application has orange and brown colors incorporated into it, to blend in with the RIT System.

7. Search: User will be able to search for tasks based on name or due date. This will be implemented using XQuery and XSLT. Java uses Saxon API to query the user's xml file and return the XML results. The resulting XML is passed through java's built in XML transform function to stylize the results.

XML Related Technology:
1. XSD/XML-Schema:
   o XML Schema is used to generate the structure for xml documents for both the to-do list and the user logins.
2. XSLT:
   o XSLT is used to style the results returned from the XQuery Search function
3. AJAX:
   o Ajax is used to add, edit or remove tasks so that the page isn't refreshed when these actions are performed
4. Security:
   o XML file of users' credentials will be used for authentication.
5. XQuery:
   o XQuery is used behind the scenes of the search function to query the xml document to retrieve information about the to-do list and the users.

6. JAXB:
    o Java API for XML Binding is used to convert xml schema into Java classes and futhermore, it is used on the fly basis to unmarshall XML data files into Java objects as well as to marshall Java objects into XML data files..
7. JSP:
    o JSP is used to add HTML tags for formatting after the task objects are retrieved from the XML file

Research Paper Connection:

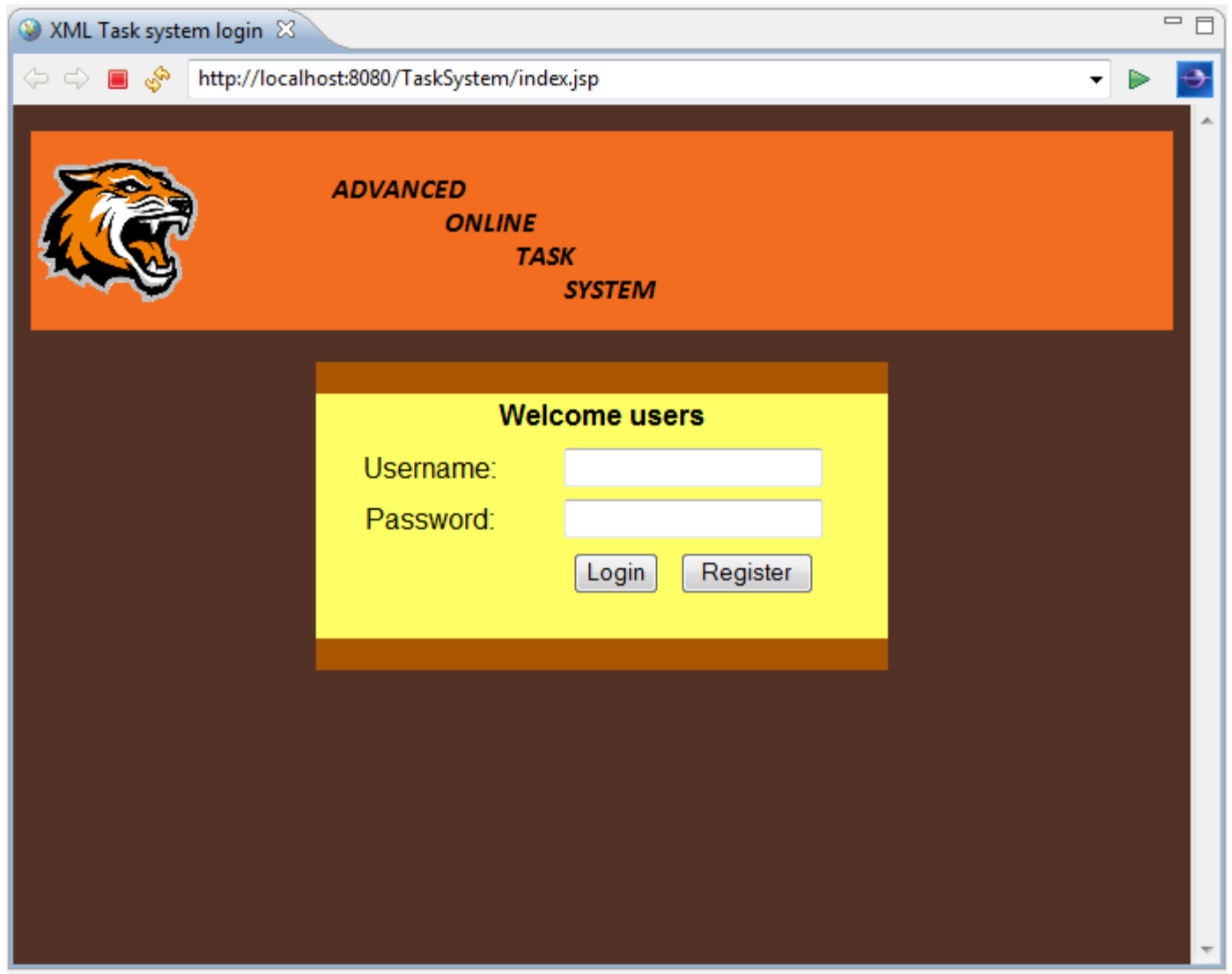Enabling SchemaFree XQuery with meaningful query focus

Authors: Yunyao Li Department of EECS, University of Michigan, Ann Arbor, USA 48109
Cong Yu Department of EECS, University of Michigan, Ann Arbor, USA 48109
H. V. Jagadish Department of EECS, University of Michigan, Ann Arbor, USA
48109

This paper introduce an idea of Meaningful Query Focus (MQF) for finding related nodes in an XML document that enable users to design schema free XQuery. We plan to use this idea for implementing Search feature of our application. With schema-free XQuery, user doesn't need to be aware of the schema of a XML document. Users will be able to query XML data with whatever schema knowledge they have. Even if they do not know the schema at all , they will be able to query XML data. The exact algorithm is not implemented in our project, but the over-arching idea and the goal of the research paper was implemented through our keyword based search that uses Xquery in the background.

Screenshots:

Log In:

Main Page:

Update:

Update-not-allowed:

Search-by-name:

Search-by-date:

Readme:
The project can be checked out and browsed online at http://code.google.com/p/jsp-xml-task-system/

To run it, the project needs to be imported into eclipse j2ee 3.6 or higher. After the project is imported, the blank users.xml file, given in files_to_be_placed_next_to_eclipse.exe folder needs to be placed in the eclipse installation directory.
Tomcat needs to be setup in eclipse if it is not already setup, and the project can be run by right clicking on index.jsp and choosing "Run on server". First step is to register a user within the application and the system is ready to be used.