

CS420 - Spring 2006

Team Assignment 1

The Geospatial Registration Project – Iteration 1

Due Date: February 14, 2:50 p.m.

Points: 200 points

Objectives:

1. Submit a Gant Chart that showing your planned activities. (Hopefully you create this before you start with this iteration of the project.)
2. Both teams should meet and agree upon requirements presented in class as well as in this assignment. Both teams should work out an agreed upon design and a remote interface based upon the agreed upon requirements.
3. Develop and test a prototype that provides end to end connectivity (with minimal functionality).
4. Create a team website on bitterroot.vancouver.wsu.edu.
5. Use the Subversion repository on bitterroot to track all documents and code.

Project Description:

Create a web based tool for creating custom Geospatial Registered content that can be saved, edited, and presented interactively to users within various privacy domains. Geo Registered content implies that we be using maps to associate user data with location. We will use a third party mapping API to generate the geospatial content. We will allow users to add their own data to the maps and allow them to save that data and the associated locations for future viewing. **Thus we will need to track and maintain user accounts.** Privacy domain implies that the users can specify who has access to their content.

Project Requirements:

1. The system must be scalable in order to perform well under heavy user load.
2. The system must be secure and reliable.
3. The system must use open source tools, but ownership of the created content including code and documentation must be retained by the University, The Instructor, and the Students.

Iteration 1 Use-Cases:

1. Display Main Webpage

Actor: Web User

Pre-Condition: Navigate to main website location using a URL in web browser.

Main Scenario:

1. Provide the user with a description of the website, a logo, and a user account login, and the option to create an account.
2. If the user enters a username and password and clicks the "login" button log them in using Use-Case 3 (User Logs into System).
3. If the user clicks on the "create an account" option take them to Use-Case 2 (Create User Account).

2. Create User Account

Actor: Web User

Pre-Condition: Use-Case 1

Main Scenario:

1. Ask user to provide a user account name and a unique 6 character password (enter password twice and make sure is not displayed using clear text). In addition, ask user to provide email address, full name, etc. Make sure that this form is extensible.
2. Check the account name for uniqueness against existing account names.
3. If the account name is not unique tell this to the user.
4. If the account name is unique make sure the passwords match, create an account, transfer them to the main interface (Use-Case 4) and track the user (using httpsessions and their unique username).

3. User Logs into System

Actor: Web User

Pre-Condition: Use-Case 1

Main Scenario:

1. Check the username and password against the existing database.
2. If the username and password is incorrect return them to the main login page and describe the error on the main login page (Use-Case 1).
3. If the username and password is correct transfer them to the main interface (Use-Case 4) and track the user (using httpsessions and their unique username).

4. Display Main Interface

Actor: Web User

Pre-Condition: Use-Case 1

Main Scenario:

1. Display a list of menu items (for now they will not have functionality except for a logout option) including a log out option.
2. Display the username.
3. If a user clicks on a menu item, execute the use-case associated with selected item (e.g. logout).

5. User Logs out of System.

Actor: Web User

Pre-Condition: Use-Case 4 (User has clicked on the log out option).

Main Scenario:

1. Allow the user to logout (this ends the httpsession and returns the user to the main log in page in Use-Case 1).

Deliverables for Both Teams:

1. Based on our discussions in class and assignment requirements, create a basic SRS that provides enough detail to carry out this iteration (description, requirements, use-cases, etc). Remember, this is only the first iteration, thus the SRS will be sparse. Each iteration will bring change and more detail. Use the template provided on the class website.
2. Use the requirements to create a simple UML design of the system. An Activity Diagram and Class diagram are sufficient. (Use Rational or Visio to create the UML – both teams must agree upon tool.)
3. A test plan (not necessary JUNIT tests) must be created and executed.
4. Each team member must review their peers using the peer review forms. Peer reviews are kept private and must be submitted via the class website.
5. Use an ant file to build and test.

Deliverable Specific to the Client Team:

1. Create a web accessible user interface for the project that complies with the requirements and Use-Cases. This should include JSPs to server up dynamic

- functionality. The content should be deployable into the JBOSS application server using a "war" file.
2. Use the EJB's provided by the server team to save and retrieve information.

Deliverable Specific to the Server Team:

1. Create one or more EJBs that provide functionality for tracking, editing, and saving user information as described by the requirements and use-cases. The content should be deployable into the JBOSS application server using a "jar" file.
2. Create a simple database that stores user information.

Submission and Grading:

1. All submitted documents will be graded on content as well as grammar. All documents must be in an html format, so they can be posted on a website if necessary.
2. Create a plain text README file that describes how to deploy and run the system. This should be placed in the docs directory and labeled (README_TH1).
3. Submit all documents and code by checking them into subversion repository. (I will checkout a version when the assignment is due.)
4. Each team must submit extra documents containing the following artifacts:
 - a. Group meeting time(s) and duration. (post this on your website). You can use a password using htaccess.
 - b. A description of who worked on what (place this in your peer review forms).
5. Each student must submit (via the website) a peer-review for each of their teammates. Use one-peer review form, zip it, and submit it to the class website.