CS1530 SPRINT-1 DELIVERABLE

Project:	Webfreesurfer
Group:	Open Science Grid - 2

Due Date: **29 SEPTEMBER 2015**

Group Member	**Role**
WIlliam McKibbin	Project Manager
Phillip Washy	QA/Dev
Robert Colleran	Developer
Nathan Dorman	QA/Dev
Zane Hernandez	QA

User Story Backlog

The format of the user stories is as follows:
As a <role>   I want a <feature> so I can <Result>
The user stories are broken into a hierarchy where larger feature requests are broken into smaller sub stories. Additionally the stories are ranked by their importance. In this case, logging into the website has been characterized as the most important, followed by data submission and so on. The sub stories are ranked as well.

1. As a **user**, I want to be able to log into the website so my freesurfer job data and records are secure.
    a. As an **administrator**, I want the users to be stored in a Postgre SQL database with a schema of my choosing so users can be authenticated with and can have additional information stored about them.
    b. As an **administrator**, I want to be able to access and modify account information of users within the site so I can support and audit the users of the site.
    c. As a **user,** I want to be able create an account on the website, and modify my information after my account is created so I can keep my account up to date and secure.
    d. As an **administrator**, I want users to have to confirm their email address via a dynamically generated link upon creating an account.
2. As a **user**, I want submit brain scan data to a website so I can have the data be processed via a freesurfer workflow on the Open Science Grid
    a. As a **user**, I want to be able to submit a freesurfer job in a few clicks so I don't have to worry about the technical aspects of running the job on the OSG.
    b. As an **administrator**, I want job data to be stored in a Postgre SQL database with a schema of my choosing so a record of past and present jobs can be monitored and audited.
    c. As an **administrator**, I want brain scan data to be anonymized and I want the user to confirm this so no confidential medical records are exposed.
    d. As a **developer** I want want the freesurfer jobs to be submitted, monitored, and deleted via a RESTful Api call to a middleware named OSGConnect
    e. As a **power user** I want to be able to use a CLI to submit, monitor and delete jobs so I can test the software and have increased functionality.

    f. As a **user**, I want to be notified via Email when my job has completed and be provided a download link to view the results so I can have better awareness and access to the results.
    g. As a **user**, I want to be able to monitor the progress of a job and be provided statistics about its runtime so I can be aware about the status of my job.
3. As a **developer**, I want to be able to easily understand and extend the functionality of the website so in the future I can add new workflows and features to the website.

Test Outline

General/Front End (low priority)

The website's design should be uniform across all pages. All inputs (buttons, text boxes, check boxes, etc.) must be functional and responsive. All forms should submit only if the required inputs are filled in. The website should display correctly on all supported browsers. These things all contribute to a consistent user experience.

Login/Account Details (medium priority)

The security of user logins needs to be tested due to users' access to sensitive data. Tests should confirm that no passwords are stored in plaintext at any point, and that user account details do not appear in the URL.

Additional testing should be done to ensure that user account details are properly linked to the SQL database. Updates to user information in the database should be accurately reflected on the site as soon as possible. The application should be tested to verify that it can service a large number of users while still being performant.

The admin's ability to change user's information should be tested and it should be confirmed that regular users do not have admin abilities. The account verification links should be tested to make sure they are unique and properly match to the user account that is being verified.

OSG Integration (high priority)

The integration of the interface with the OSG will require testing of its own. The submission process of the brain scan should be stress tested for reliability over many rapid submissions and for performance when handling large files. The storing of job data in a PostgreSQL should be tested for accurate handling of jobs of all sizes, as well as for performance. The anonymization process for the brain scan data should be tested for completeness in removing identifying information. The monitoring of the freesurfer jobs should be tested for accuracy so that the user can accurately estimate a job's progress, and the latency of the monitoring process should be tested to ensure that it is as low as reasonably possible. The deletion of freesurfer jobs should be tested for completeness, so that no old data could clog the database overtime or be unintentionally accessible. The process of emailing users about job progress and results should also be tested for reliability.

Decision Descriptions

Prioritization

For our project, we are were given a set a fairly well defined and detailed set of specifications with prioritization baked in. This made it fairly easy to prioritize certain features over others. The specification detailed the basic use cases for the software as follows:

1. Account creation.
2. Account maintenance including password change/reset, profile changes.
3. Upload source image sequence.
4. Download results.
5. Fetch job status.

This ordering is reflected in our user stories prioritization. The rationale behind this priority is fairly simple. A user must be able to be created within the system before any data upload/storage functionality is added because all data is sensitive and user specific. Once a user can be added and maintained in the system, image upload and job creation for users can be added. Once a job can be created and processed, results can be stored and given the users.

The clients also differentiated the core functionality of the application from enhancements that were secondary. Core functionality was comprised of user login and authentication, image processing, CLI, user notification, and result download. Enhancements were categorized as monitoring and statistics, multitasking, enhanced process interfacing, and dynamic web content.

Customer Interactions

Meeting with our project's point of contact, Dr. Krieger, gave us a different perspective on what we were creating and who it was for. The typical user of this website may be processing complex brain scan data and have technical knowledge, but wants to have a total abstraction of processing details. Dr Krieger was adamant that the user interface be extremely easy to use. He wanted to see jobs being created and processed with a few clicks. He noted that this software could potentially be used by thousands of researchers. He also prioritized having a functional prototype as soon as possible so he would have something to show during a presentation to the OSG in the coming weeks. Another priority that he expressed was that the software be easily extended so that later on, after we had left the project, more functionality could be added.

Suchandra Thapa, our technical contact, provided us with the MVP specifications that answered nearly all technical related questions. There was some confusion about how our software was going to utilize the OSG but after speaking to him on the phone our role in creating this system was very clear. He is taking responsibility for creating, maintaining and processing jobs on the OSG via a middleware called OSGConnect. Our job is to create a front end for this service and maintain the users of it.

Background Research

Once our understanding of the project requirements matured, we were able to narrow down the early design decisions that needed to be made. Most aspects were decided by the clients and were given in the specifications. The only big decision left for us to make was what backend framework to use for our site. Since the class requirements specify that java must be used, it narrowed our search significantly. Our team's experience with web frameworks was fairly limited so we needed to take this into account before deciding. One member of our team had experience with a java MVC framework call JSF(Java Server Faces). After considering a few other options we decided to go with JSF. Given a good resource for help on the team and with JSF's modularity and extensibility, it seemed to be the best option.

Ineffective

Our group suffered from some communication problems early on. We had trouble finding times to meet up and discuss the project as well as establishing reliable lines of communication. These problems remedied themselves as the deadline for sprint one approach and it became necessary to prioritize the work. This is something we plan to stay on top of as the next sprint approaches as the level of work is going to increase significantly

Effective

Our group was effective in analyzing the requirements and specifications laid out by the clients. We have a clear vision of the work that needs to be done and how to do it.