Specifications Document – Alpha

Product Overview

This product is intended to assist users in the rescheduling of specific sections.

Requirements

Priority Definitions

1. This requirement is a "must have" for the product to function.
2. This requirement will make the product much more efficient, but is not necessary for it to function. It is still highly imperative.
3. This requirement is a luxury and is not necessarily needed. Functional Requirements

Functional Requirements

Function	Priority	Date Reviewed	Approved by
The system should provide all information available via Banner regarding the rescheduling of the section in question for the user to view.	1	8/12/14	Jonathan Nix
The system should provide the optimal solutions for rescheduling a section.	3	8/12/14	Jonathan Nix
The system should provide information in order of relevancy. The user determines relevancy and chooses which information they want to view.	2	8/12/04	Jonathan Nix

User Interface Requirements

The user interface should…

- Provide a user login screen.
- Provide prompts for the user to input current section information. Instructions on what input to give and what to do to run the application should be clear.
- Provide output as information in a table format. This will look like the already existing room times' table.
- Provide output as information in a list format. An example of this is displayed below the same existing room times' table.

Usability

The product should…

- Be easy to learn and understand. Clear instructions will be provided on each page regarding how and where to give input and how to view different output. (See User Interface)
- Only be useable by authorized personnel. (See security)

Performance

The product should…

- Be able to support multiple users at once.
- Be able provide correct feedback within an average of 5 seconds.

Manageability/Maintainability

The product should…

- Have access to Banner and download information each night.
- Run on multiple at least two servers so that if one fails, the system can remain operational
- Have embedded Ruby files that allows our system to be more easily maintained.

System Interface

- Our system will use MongoDB.
- Our system will use the testing framework, Cucumber.
- Our system will use Sinatra.

Security

- User will create a username and password. That information is submitted to our database. Our system will hash and salt the password. We will then store the hashed password along with the username.
- Our system will check to see if the username already exists.
- Our system will make sure the password meets requirements. (i.e. at least six characters long)