Team Zaboo:
Raj Patel
Jessica Lin
Landon Gray
CS374-F14
10/27/14

Project Verification

I. **Black Box Testing:**

Feature: The user should be able to log into the system.

Scenario: User submits correct username and password
Given user is on login page
And user types correct username
And user types correct password
And user clicks the login button
Then the user logs into the system
Then scheduler view page should display

Scenario: User submits incorrect username
Given user is on login page
And user types correct username
And user types some password
And user clicks the login button
Then page returns invalid username or password error

Scenario: User submits incorrect password
Given user is on login page
And user types some username
And user types correct password
And user clicks the login button
Then page returns invalid username or password error

Scenario: User submits password and no username
Given user is on login page
And user types username
And user clicks the login button
Then page no specified username error

Scenario: User submits username and no password
Given user is on login page
And user types password
And user clicks the login button
Then page no specified password error

Feature: The user inputs CRN,requested time, days (Monday,Tuesday,etc.) and sent to database

Scenario: Successful storing input from webpage to PHP file
Given there are no current schedule conflicts with inputted CRN
When I inputted 123456, 10:00-11:00, Monday (checked), Wednesday (checked), Friday (checked)
Then the variables (CRN, beg_time, end_time, days[i]) should have

123456, 10:00,11:00, [Monday Wednesday Friday] respectively

Feature: Finding Conflicts

Scenario: Data from PHP conflicted to time data in database
Given: data from php
When: there is a time conflict
Then: time that is conflicted is stored in a PHP variable

Scenario: Data from PHP conflicted to room_num in database
Given: data from php
When: CRN class room number conflicts to another CRN class room number
Then: room number that is conflicted is stored in a PHP variable

Scenario: Data from PHP conflicted to days in database?
Given: data from php
When: there is a day conflict (no time availability on that day)
Then: days that are conflicted are stored in a PHP array

Feature: user inputs CRN, beginning and end time, days of the week to test

Scenario: No conflicts with student schedules
Given: database of student schedules
And: course catalogue
When: user inputs course number and testing time and day
Then: new screen appears dictating no conflicts with the option of viewing the calendar grid of severities
And: The time block specified will be outlined and shaded white to dictate no severity

Scenario: Conflict with senior's schedule, not needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded yellow to dictate a low severity

Scenario: Conflict with senior's schedule, needed for graduation, offered next semester student will attend school
Given: database of student schedules
And: course catalogue
And: catalogue year
And: graduation date

When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded orange to dictate a high severity

Scenario: Conflict with senior's schedule, needed for graduation, not offered next semester student will attend school
Given: database of student schedules
And: course catalogue
And: catalogue year
And: graduation date
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded red to dictate the highest severity

Scenario: Conflict with junior's schedule, not needed for graduation
Given: database of student schedules, course catalogue, and catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red. The time block specified will be outlined and shaded yellow to dictate a low severity

Scenario: Conflict with junior's schedule, needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded orange to dictate a high severity

Scenario: Conflict with sophomore's schedule, not needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded green to dictate a very low severity

Scenario: Conflict with sophomore's schedule, needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded yellow to dictate a low severity

Scenario: Conflict with freshman's schedule, not needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded green to dictate a very low severity

Scenario: Conflict with freshman's schedule, needed for graduation
Given: database of student schedules
And: course catalogue
And: catalogue year
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded yellow to dictate a low severity

Scenario: Conflict with multiple students
Given: database of student schedules
And: course catalogue
And: catalogue year
And: graduation date
When: user inputs course number and testing time and day
Then: new screen appears with calendar grid, with every grid either white, green, yellow, orange, or red
And: The time block specified will be outlined and shaded the color of the highest severity amongst the students

II. **Module Testing:**

A. **Sign-in-Testing-** Tests sign-in page if the sign-in data is submitted to the PHP files
B. **CRN-sending-Testing-** Tests if the CRN data got sent to the PHP files
C. **Color-Coded-Testing-** Tests the Google Calendar for appropriate color of

certain conflicts displayed on it. The testing fails if the Calendar has a wrong color for a certain conflict.

III. **Integration Testing**

A. **Log-in Testing-** Tests the log-in PHP files for logging in attempts. If the user enters the correct username and password, then he/she is taken to the index page. If the user enters the incorrect username or password, then the page would display “Incorrect username or password” and the number of attempts would decrement.

B. **Data-Input-Validation-Testing-** Tests the PHP files that connect to database for data validation (CRN) in the our current database. If the test can find the section with the CRN, then it would print “OK”. If the test finds the CRN even though it’s not in the table, then these PHP files would fail the test.

C. **Conflict-Search-Testing-**

1. **Time-Conflict-Search**: Tests the conflict searching PHP file for time-conflict findings. With the inputted CRN time data, if the PHP file successfully finds the time conflict with another CRN in the database, then this file passed.
2. **Days-Conflict-Search:** Tests the conflict searching PHP file for day-conflict findings. With the inputted CRN days, if the PHP file successfully finds the day conflict with another CRN in the database, then this file passed

Note: If the PHP file finds a conflict even though there isn’t one, then it failed both tests automatically. Also if the PHP can’t find a conflict although there is one, then the file fails the test.

D. **Conflict-Calendar-Display-Testing-** Tests if the conflict-storing PHP file displays all conflicts on the Google Calendar. It fails the test if not all conflicts are displayed.

E.

IV. **System Testing**

System testing will be done by as a result of other testing. For example part of the system testing includes black box testing so we will use the cucumber code for this. We will also work to implement a secondary server incase our main one fails. We can test that the system stays online when one of the servers goes down.

V. **Acceptance Testing**

Once the modules have been integrated and system testing is done, we will bring the project to an administrator, probably Karen or Dr. Reeves. They will then be given the chance to test out the system themselves and see if the functionality and aesthetics are up to his/her standards. In other words, do they like the appearance of the web based application and does it output the information they expected in a legible manner that would make utilization easy. If yes,

we've done our job. If not, we take it back and add in whatever the admin felt was missing or change what they didn't like and then release product Zaboo 2.0 and do it all again. Ideally, this testing portion would be done the last week of November, no later than November 28th, giving us a week to make any changes post-testing before the final product is due.

VI. Performance Testing

This portion of testing will most likely be done after the system integration, either during or immediately after User Acceptance Testing. If we're feeling like overachievers, we would test the performance of modules and then test them after they've been integrated. The more likely path however will be putting all of the pieces together and timing how long it takes for our application to output the calendar of conflicts. Ideally, the information would be displayed in 3- 5 seconds. That's about how long other web pages take to load and ours should be held to the same standard. If it takes longer than 10 seconds to load, it would safe to say that we need to rework our processes/function calls and maybe even re-analyze our database setup by going our ERD Diagram and DFD's. Theoretically, this would be done by November 30th, giving us time to try to make it more efficient but recognizing that a functional product that is a little slow is better than a late product, or even worse, a lack of one.

VII. GOMS Testing

Keystroke Level Model

Tester: Our Team and one other user.

Press a key or button

Best typist = .08 seconds

Good typist = .12 seconds

Average skilled typist = .20 seconds

Average non-secretary = .28 seconds

Typing random letters = .50 seconds

Typing complex codes = .75 seconds

Worst typist = 1.2 seconds

Point with a mouse (excluding click) = 1.1 seconds

Move hands to keyboard from mouse (or vice-versa) = .4 seconds

Mentally prepare = 1.35 seconds

Goals:

Task: User types username and password and clicks login button

- **Task groups:** User prepares his or herself,User moves, mouse types username and password

  Task: User types in crn and enters a time

- **Task groups:** User prepares his or herself, User moves, mouse types username and password

VIII. **Status Reports**

Simply checking if weekly status reports were submitted with actual information on the progress of the project. For example, week 1, the front end was designed. Week 2 and 3, database created and tested. Week 4 was integration testing. Providing visual proof of the tests in the status reports would be bonus.