The OSCAR Redesign Project
Part 3 System Prototype and Evaluation Plan
CS3750 User Interface Design

The Grouches:
Jared Salzmann
Jack Gruendler
Jason Tongco
Carlos Rios

Table of Contents

Introduction

This OSCAR redesign is, first and foremost, not an overhaul of the backend. Such changes are beyond the scope of this project. This redesign is focused upon the user experience and placing within easy reach of the user the most-used or most-requested functions.

Requirements Summary

First and foremost, the redesigned OSCAR should have easy access to the functions students use frequently. Specifically, looking up classes, registering classes, and viewing schedules should be accomplished with a minimum of mouse clicks. It should be easy to navigate for the first time without the help of FASET volunteers. It should integrate more powerful search features, such as searching by GPA, and display features, such as plotting classes on a campus map. Display of information should be better organized: the user shouldn't have to look at several different pages to find out everything about a class (in contrast to the current system, which requires the user to look at the search results, click on the CRN for the class, then click “View Catalog Entry” to see all information). It should be more customizable and intelligent, by having persistent user settings (for things like search term and campus), with reasonable defaults (such as “current semester” and “main Atlanta campus”). It should not break when pages are opened in multiple windows.

Final Design Summary

We chose to prototype our Calendall design from phase 2. Many features and interactions have been fleshed out and redesigned. The goal of this design is to help the student organize their classes and other school activities in a visual way on a weekly schedule. Calendar systems are fairly common and most students are familiar with them. A visual calendar provides direct manipulation of classes as they relate to time. The calendar also allows excellent visibility because it shows the classes that are registered and the most common class conflict students have (time conflict). With the use of a simple fuzzy search instead of advanced search parameters, this design eliminates clutter. As we found in research, students spend a lot of time deciding which classes they want to take before they are aloud to register. To help the students in this situation, the calendar system has the ability to have multiple calendars to work with at once. Students can create multiple schedules that they would like to take. When it is time to register, the student can see which of the previously created schedules has all the classes open. The student then registers for all classes in the schedule with one click of a button. We felt this is the best design to prototype because it applies to all students. This one design could service people of all majors and isn’t based on major requirements that would be different and complex for every major. A class planning/major requirements program could be added on top of our calendar system (such as the one being designed by the “Get Me Out of Here!” team).

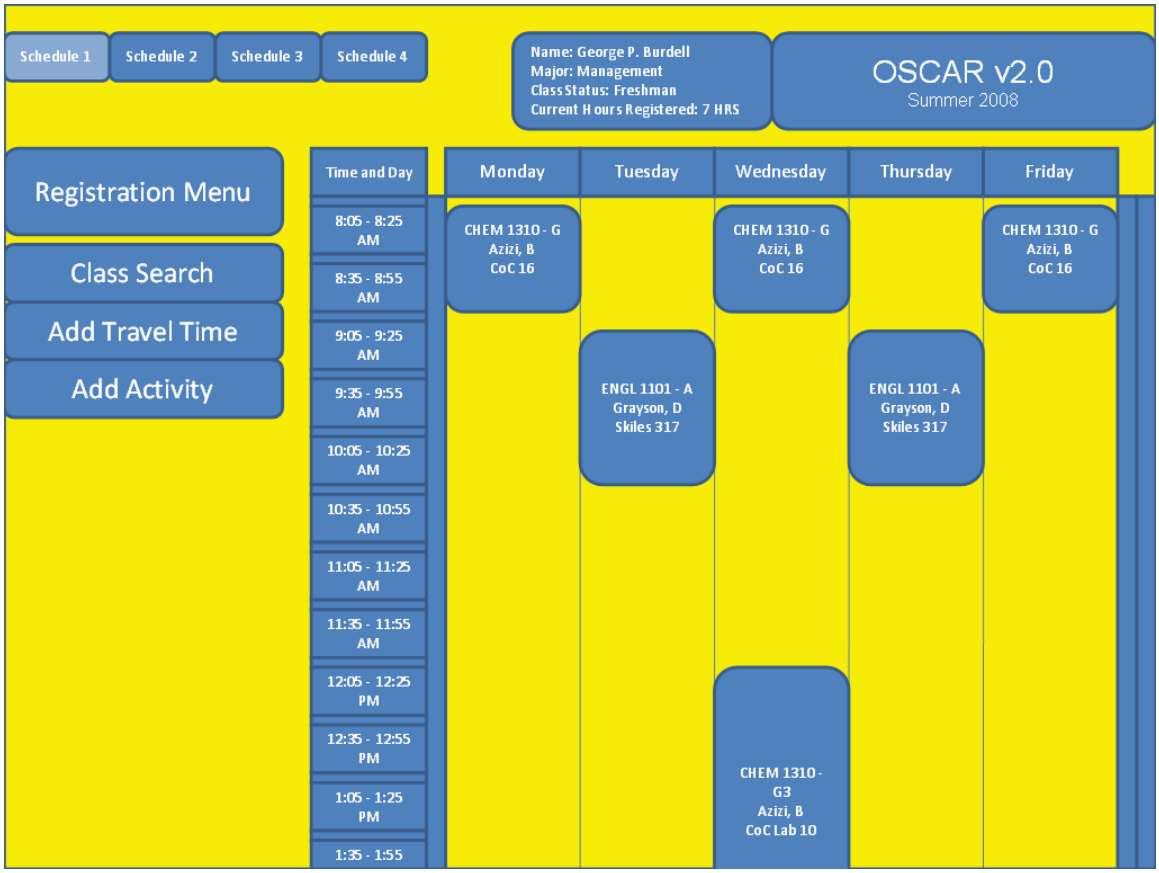
Insert Picture from Carlos' Prototype

Prototype Description

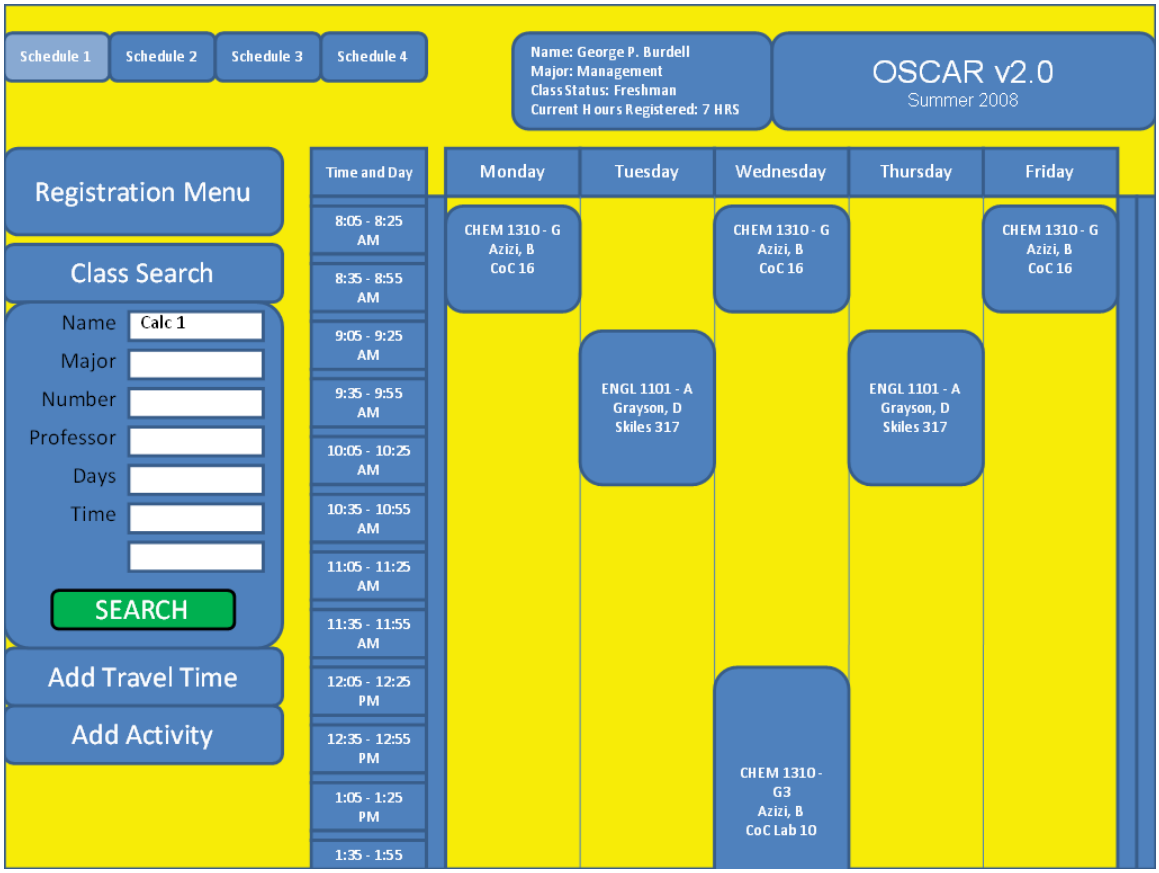
Our prototype is a semi interactive PowerPoint. We were unable to create a fully interactive PowerPoint because of the limits of PowerPoint (no complex animation etc). None of us has any experience with Flash and we felt a code/software based prototype would require more time to complete than we had. Our PowerPoint was very easy to develop the user interface without having to deal with any backend. The main limit of PowerPoint is that each state of the system can not be parameterized so each unique state needs its own new slide. Because there are hundreds of states in a fully featured version of our design, our prototype will be limited on features and complexity. Our prototype is a medium fidelity prototype, in that users can interact with the interface but in a limited and structured fashion. The data being displayed (classes) are all going to be static and simulated. We attempted to make the highest fidelity prototype that PowerPoint allows and the result is a scripted simulation. The users will know that the interface is simulated due to a number of PowerPoint issues. Instead of being able to type in a class to search, our prototype simulates a class being typed in. PowerPoint also shows a pointing hand on every object. In addition, our prototype is not able to cue the user to what objects are clickable by use of hover-over changes.

The basic usage actions that we simulated in our prototype are as follows. Screenshots are given to enhance the readers understanding.

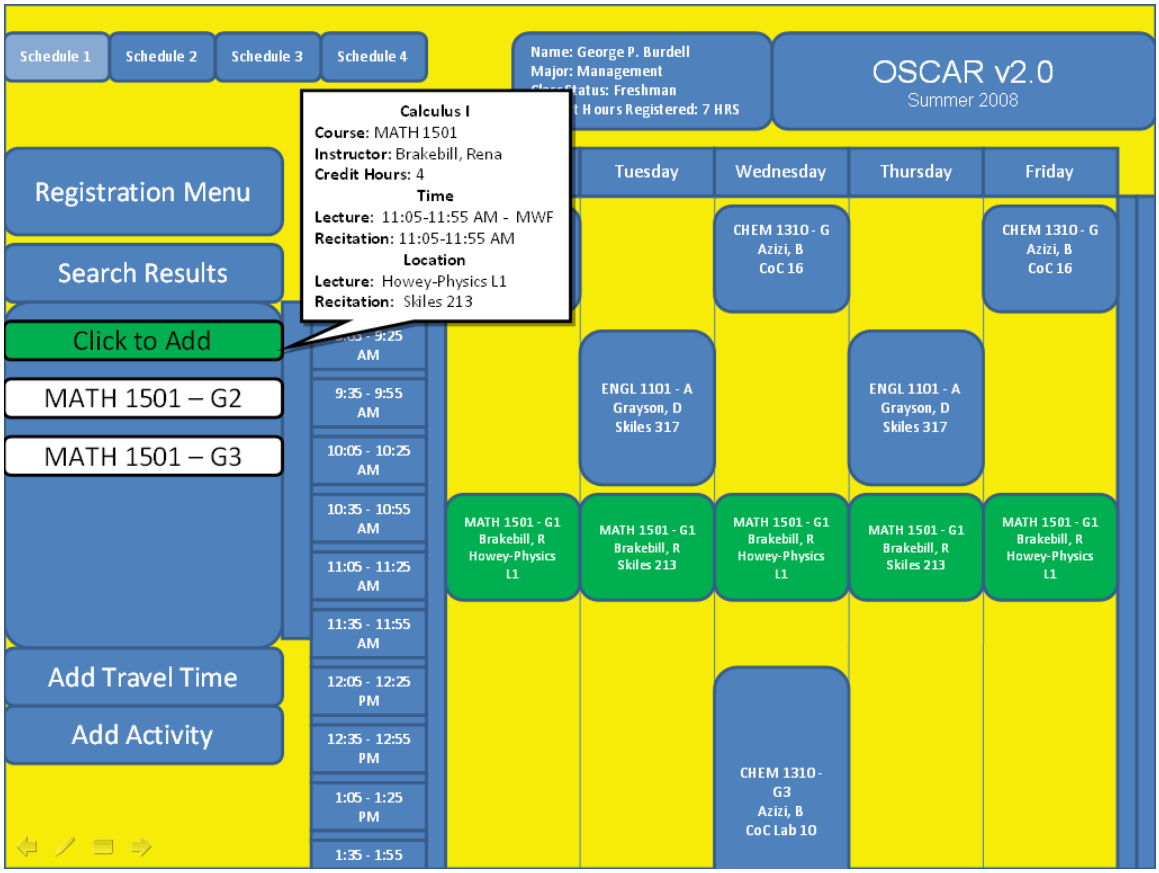
The main display is shown here. When the user puts their mouse on a class on the schedule, detailed information about the class is shown in a small popup box.



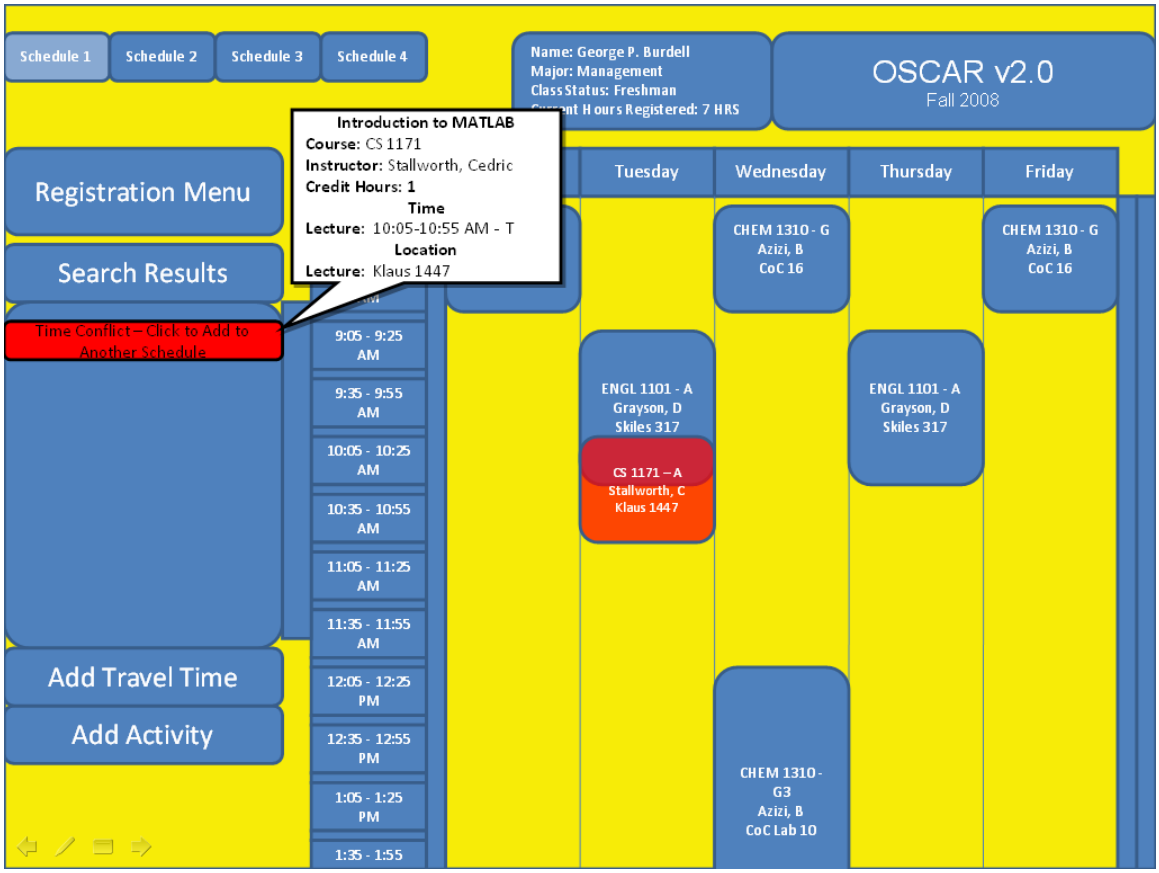
To search for a class, the user clicks “Class Search” and a list of search options is displayed. The user will click on the search box and “calc1” will automatically be typed in (PowerPoint limitation).



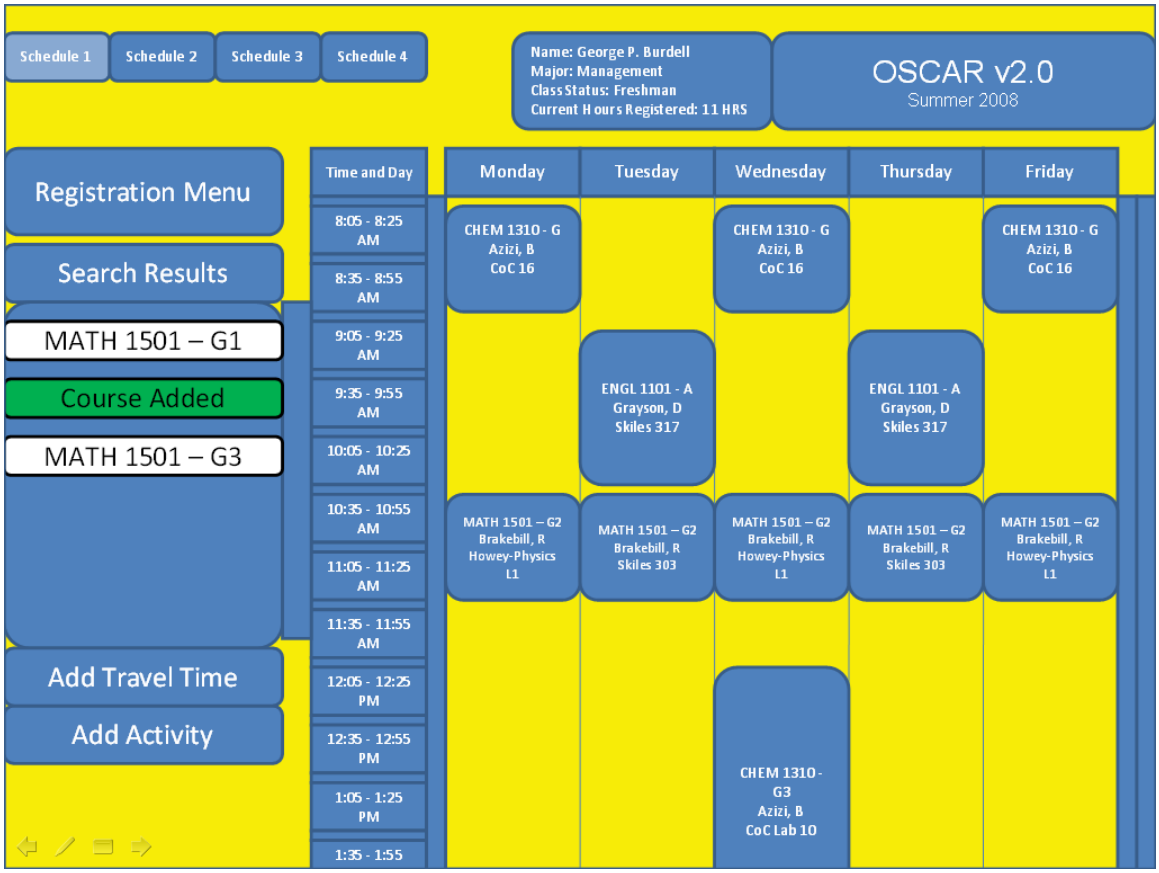
The user then clicks the green search button to search for a class. The class results are populated.



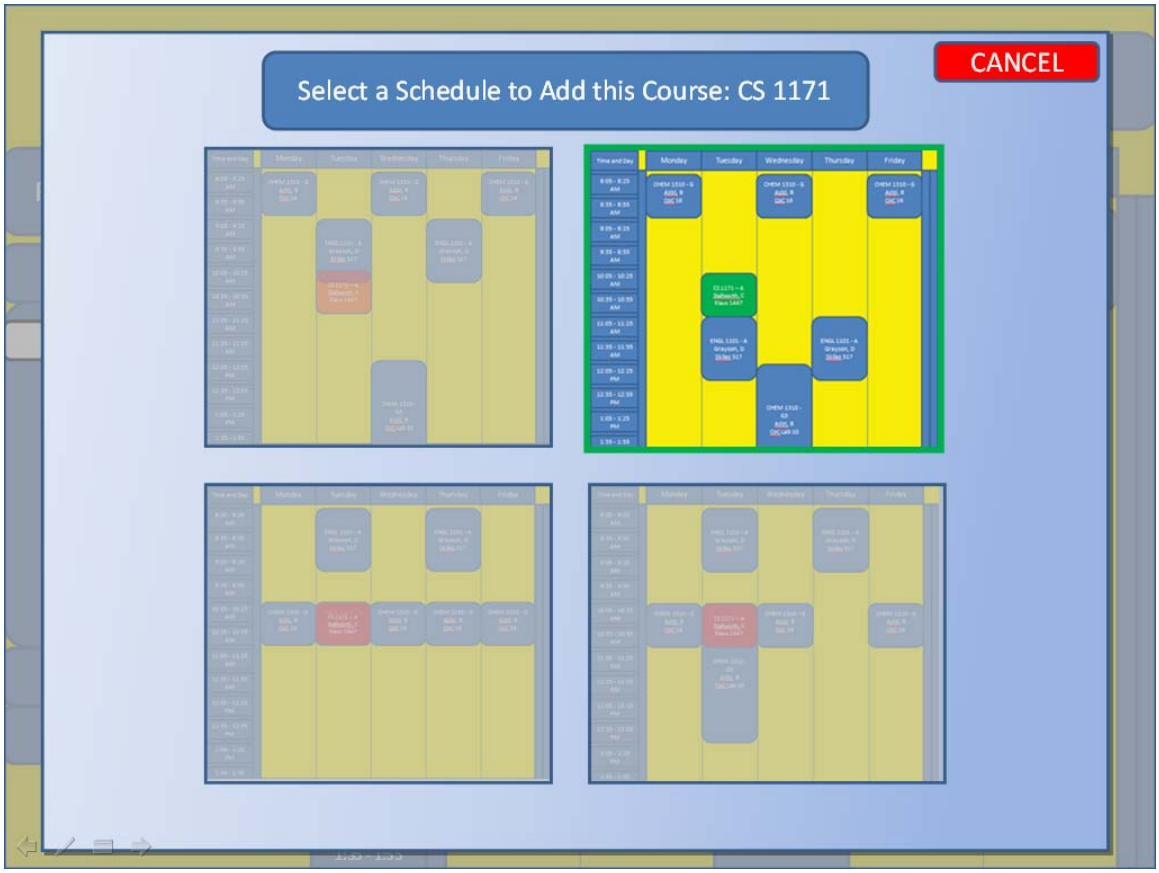
When the user hovers over a class among the results, a popup comes up showing details of the class.



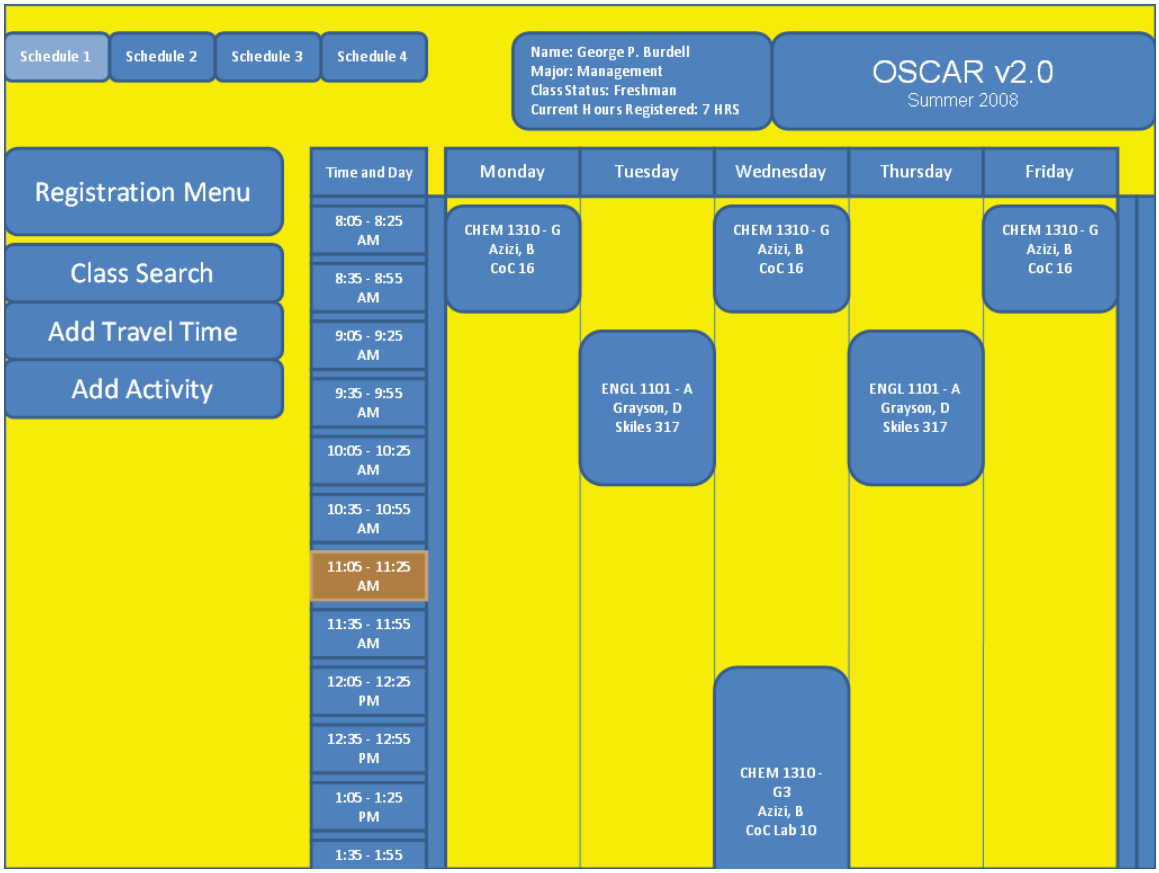
The class shows up in the calendar—as red when a conflict and green when no conflict. The button turns green or red for the same reason. The button says “Click to Add” if the class has no conflicts and “Click to Add to Alternate Schedule” if there are conflicts. When the user clicks on the “Click to Add” button, the button changes to “Course Added” and the class shows up in the schedule.



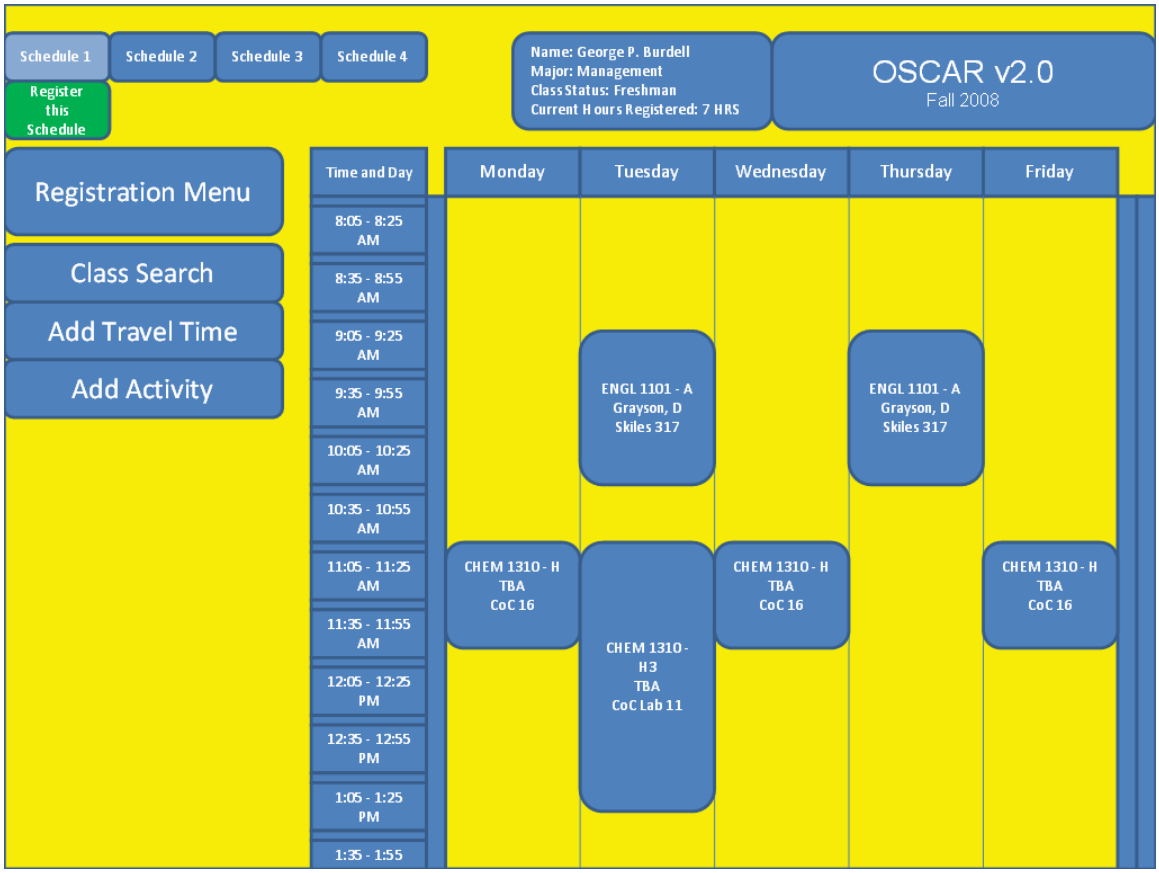
When the user clicks on the “Click to Add to Alternate Schedule” button, a popup comes up, taking over the entire screen.



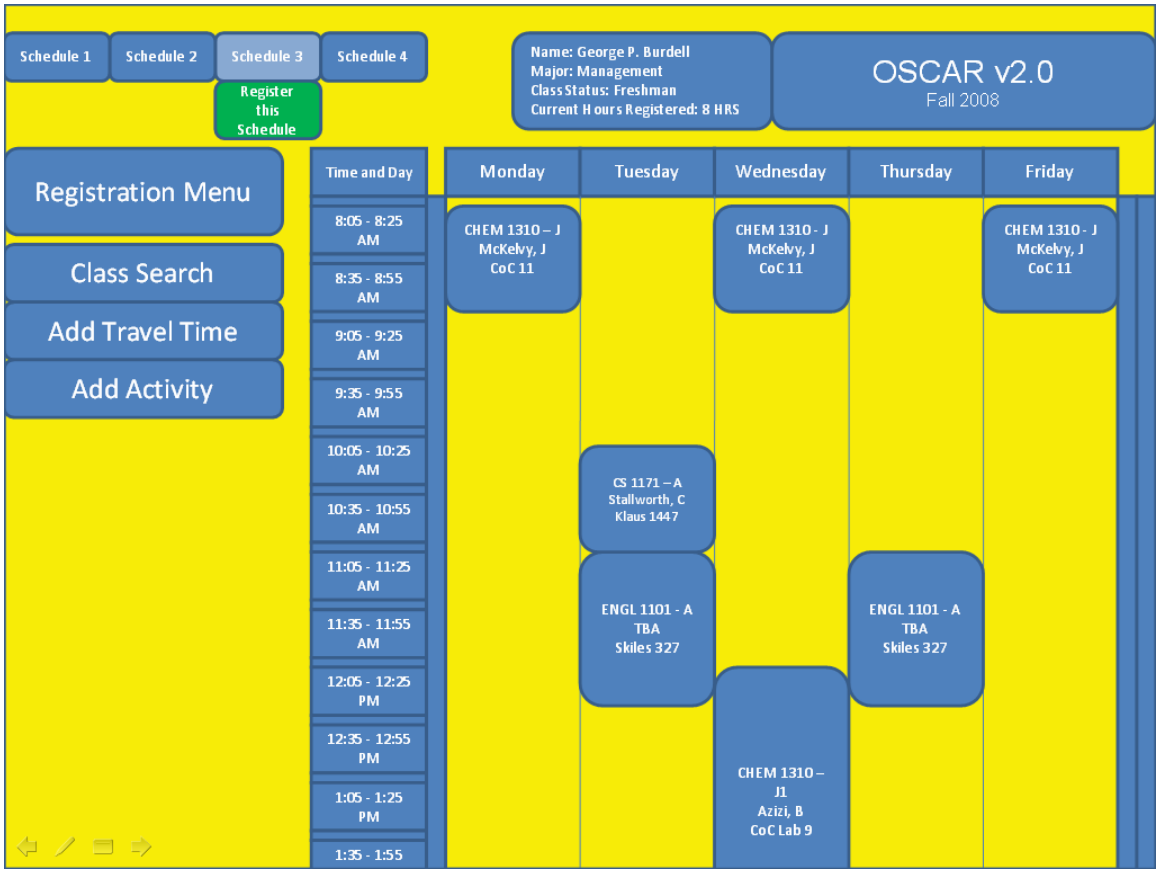
The class in question is shown on all four alternate schedules (zoom out mode). The classes are red if there is conflict and green if not. If green, the schedule becomes a large button that the user can click on to add to that particular schedule. There is also a cancel button that closes the large popup. If the user clicks on a mini schedule, the class is added to the schedule. The user is moved to the alternate schedule where the class was just added. The search area remains the same except the class entry in question says “Class Added”. In order to search by time, the user clicks on a time slot in the calendar.



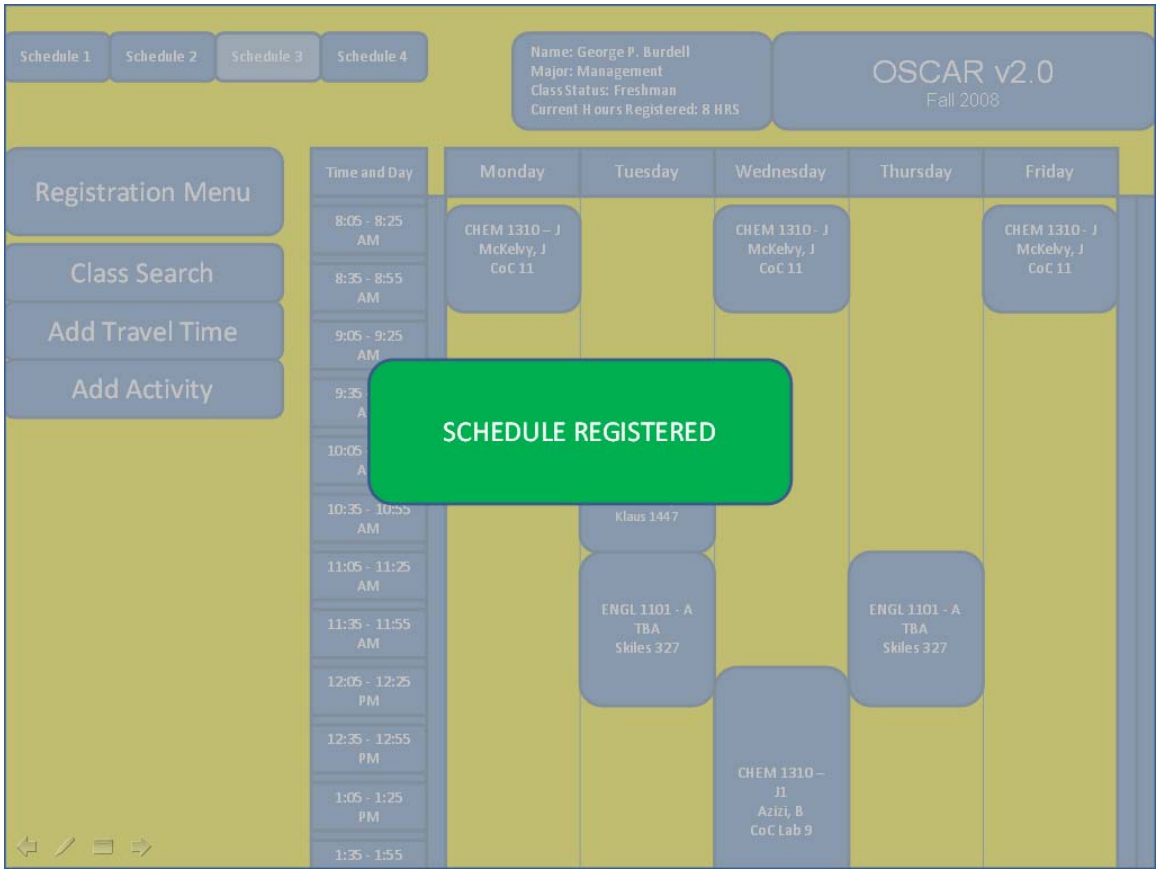
If the user selects, for example, 9am, the class search results will display all classes that start between 9am and 9:30am.



The alternate schedule tabs at the top left of the screen show the four different tabs. The lighter tab represents the “selected” tab. The user switches between tabs by clicking on the tab they want to switch to.



Only the schedule changes when switching between tabs (not the search area). When it is registration time, a “Register this Schedule” button is displayed on the tab that is selected (and light).



When the user clicks the button the users is told that the registration is successful.

Evaluation Plan

We will use our prototype to evaluate the following features:
searching by class name
searching by time
displaying class information
adding classes
resolving schedule conflicts
creating multiple schedules

We have chosen these features to test because they're important to the core purpose of the system, and they are areas of functionality in which our prototype differs significantly from OSCAR.

We will benchmark searching for a class by name, searching for a class by time slot, and resolving a schedule conflict by videotaping the evaluation session and measuring the elapsed time for each task.

We will recruit our testers from the student body, and record their academic year and major for demographic purposes. It should be noted that the evaluation will occur during Fall registration, which is convenient because the users should have registration-related issues fresh in their minds.

To evaluate the interface, we will give the user a task to perform and then measure how much time it takes him to either accomplish it or become frustrated. Because our prototype is a scripted simulation, the user cannot explore the interface as if it were fully interactive. Therefore, actions that are not in the pre-determined task sequence will have no effect.

In addition to the benchmarks, we will issue each tester a written questionnaire at the conclusion of the test. It will have two sections, one containing Likert scale questions and the other containing open-ended, paragraph-answer questions. The questionnaire will measure how well the user believes the interface meets his/her needs. For example, it will allow the user to express his/her opinion of the effectiveness of the calendar metaphor or multiple schedules, suggest further improvements or corrections, etc. The questions will include the following:

Likert scale questions:

I liked this interface better than the current implementation of OSCAR.
I became frustrated trying to use this interface.
This interface helped me focus on the task of registering for classes.
I would like to have the "multiple schedule worksheets, accessed by tabs" feature in OSCAR.
I would like to have a calendar-centric interface in OSCAR.
I found it easy to search for classes by name.
I found it easy to search for classes by time.
I found it easy to resolve schedule conflicts.
The color scheme chosen was helpful.
I made errors when using the interface.
I was able to complete the task quickly enough.

Open-ended questions:

What aspect of this interface did you like the most?
What aspect of this interface did you like the least?
What single thing would you ADD to the interface?
What single thing would you REMOVE from the interface?
What single thing would you CHANGE in the interface?
What was the easiest part about using this interface?
What was the hardest part about using this interface?
What did you like/dislike about the search function?
What did you like/dislike about the multiple schedules/tabs?
What did you like/dislike about the calendar view?
What other thoughts/suggestions do you have?

Reflection

We learned that prototypes do not need to be high fidelity to be valid for testing. We learned that a team can create a prototype even if none of them know how to code or create a Flash program. We feel our project is going ok but we are sometimes frustrated because we can't implement all the features we want. If we had more time we would have liked to learn flash or some other prototyping program. Our team functioned fine except for our missing group member, Jason. He only showed up to the first short meeting and we haven't seen or heard from him since so he didn't help us out during this part of the project. We feel our prototype is featured enough for users to easily get an idea of how our design works and thus provide us with feedback. The benchmarks will help us understand many heuristics. Our project focus has remained the same throughout the semester. The most difficult part about the prototype was making the tough decisions to remove features due to time constraints and PowerPoint constraints. We really wished someone knew how to use a better prototyping tool such as Flash.