Cougar Scheduler

James Owens

Executive Summary

Students at Brigham Young University know how much of a pain it is to register for classes every semester. Ever since I was a freshman there have been classes that I have had a difficult time registering for. There are some really popular classes that everybody has a difficult time registering for. Usually the result ends up with the individual waiting another semester for the class or the individual checks Route Y multiple times a day hoping they will see an available spot and grab it before someone else does. Of course there are about five to ten others doing the same thing making it more a competition than anything else.

Cougar Scheduler is a program that takes the place of an individual. In a brief summary, it allows the user to login to the BYU system and allows the user to select a class they would like to add. It will then add that class or continue to try adding the class until that goal is achieved. It therefore makes it so the individual does not have to waste time checking Route Y several times a day in order to get the class they want.

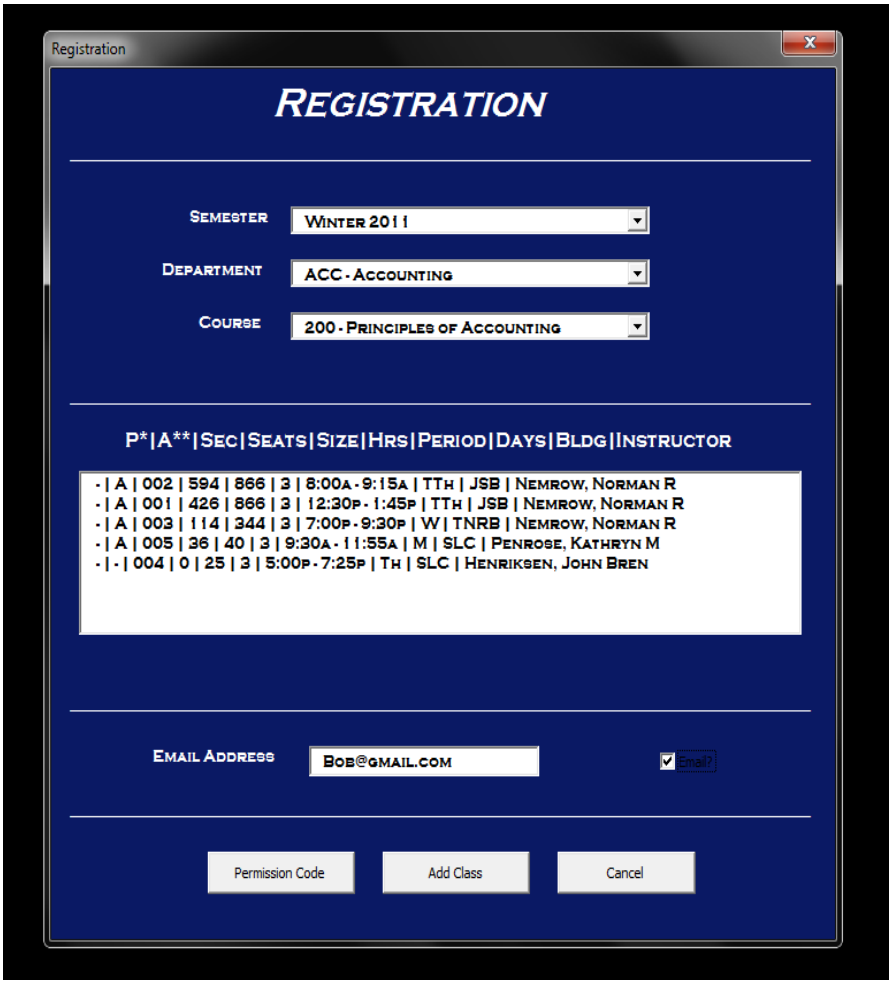
Implementation

Individuals using Cougar Scheduler will only interact with user forms. This was done primarily for ease of use. Many parts of the forms are similar to parts of the web browser and JavaScript students would interact if they were registering through the web browser like normal. This is to make them feel more comfortable and make the interaction easier.

James Owens

The program is composed of two forms. The first is a login form. This allows the user to input their Route Y information. After submission, the form checks to see if a user is logged in and logs them out if that is the case. This is just a check to make sure they aren't registering classes for somebody else who might have already been logged into Route Y on their computer. The form then inputs the user ID and password into the BYU login page, therefore logging the user in. If they input a wrong password or username a dialogue box will pop up informing the user the login information is incorrect.

Once the individual is logged into Route Y, the second form is populated. The user usually needs to be patient for the form to appear after the login button has been pushed because BYU servers can sometimes be very slow. Estimated time between the two is about 12 seconds. The second form is the primary form and is where the user is able to input information and register for their classes.



The Registration form has many aspects to it. The top part is the sequence that allows the user to executive java script and select which class they want to view. The list box below it pulls up all of the possible class combinations the user has. As you can see from the picture, it gives the following information:

- Whether the class is available and how many seats are available
- The time the class period begins and end

- Which Days the class falls under
- The building the class will be at (Mostly to identify if it is the Salt Lake Center or not)
- The instructor

The user then has a couple of options. The first is an email text box. If the user enters an email and checks the box, the program will email the user once the class has been added or if it tried to add the class and the user did not have a sufficient remaining amount of credits. The email will inform them of what the situation is. The next options are the buttons. If the user clicks on cancel, the form will close and the program will exit. If the use clicks add, the class will add. If there is no room available for the class, the program will run a loop where it continuously tries to add the class. I entered a wait time in the loop of 30 seconds. This means the program will only try to add it every 30 seconds. This is so the servers do not get continuously overloaded by users of the program and so BYU staff members do not come looking for Professor Allen. It is also important to mention a couple of items.

1. There is also a permissions box. The purpose of this box is so an individual can add a class that requires them entering in a permission code. This was not originally part of my plan nor is it likely that and individual would need to use it, so I therefore did not associate any code with it. If pushed, it does absolutely nothing. The reason it is there however, is in case somebody wanted that function and therefore wanted to add on to it themselves. They are some lines of code I created in order to help that individual if they wished to develop it. The code can tell whether or not the class requires a permission code and therefore the user would need to add the java script aspect.

2. Outlook needs to be setup for the email function to work. At first I wanted to go a similar route to the example we did in class with Gmail; however, since everybody could see the source code and I did not want people seeing personal information of mine, I thought it would be better to create an email function that would work on computers with outlook setup. This way to program works on startup and code does not need to be adjusted with private settings.
3. During the automation portion when the program is looping, if the user gets tired and wants to stop the program without having to use the task manager, the user can simply push the cancel button. This will stop and exit the program. The program will also exit on completion of adding a class.

Technical/ Difficulty

This project was a lot more difficult and required more than triple the time I originally thought it would. I am not very familiar with html or java script so it was difficult for me to conceptually understand what it was I need to do to get it to work. Below is a brief outline of difficult and not so difficult parts of the project and what I did to make things work.

Below is a list of sub procedures I feel worth mentioning:

- Login () – This procedure logged the user into Route Y and was similar to the one done in class.
- JavaSemerster(), JavaDepartment(), and JavaCourse – These three sub procedures is what allows the user to select their semester, department, and course. The difficult part for this was

going through and parsing the html text to add options to the combo boxes. It was then difficult finding the value of the item selected and submitting the java script code. Functions from the agent class were used here to accomplish the tasks. These included the execscript, moveto, position, waitforload, and gettext functions.

- GrabValues () – This sub procedure went through imported data from BYUs website and found all of the values needed to add a course. This involved many mid, len, right, and left functions to change strings and values. It also involved parsing html code. Even though it wasn't the most difficult of the sub procedures it required a lot of time.
- AddClass() – This was definitely the hardest of the sub procedures and required a lot of help from Professor Allen. After finding the values in the sub procedure above, it was easy to enter them in and submit the java script code. The hard part was automating the procedure so it kept going until the class was added. To figure out if the class was added we needed to link a new agent to the new box that popped up after the java script was executed. It was difficult to link a second agent to the new window because the new window had the same URL link as the original window. We therefore had to create a new function in the agent's class that allowed us to attach a second agent to the new window. This function was able to do add it by searching all instance of internet explorer for a specific text instead of by the URL. Once the second agent was attached to the new window, it was easier to run the loop because I could look through the html text and see if it stated the class was added or not added. This was definitely the hardest sub procedure to build.
- Email() – The last class I would like to mention was the email class. It creates a simple email that states whether the class was added or whether the person needs to free up some more credits. This was not the most difficult sub procedure.

James Owens

Conclusion

I thought about changing my project on several occasions. I did not think it was going to be as difficult or as time consuming as it was. I am happy however, that I stayed with the current project and was able to accomplish the task. It was a great experience overall. I also wanted to acknowledge a few individuals for their help on the project.

- Primarily Professor Allen for my use of his time and the agent class.
- James Perriton for the help his VBA project from earlier gave me.