

The Olive Garden Scheduler

Joshua Tolman

December 9, 2010

Isys 520

Executive Summary

Description of the Business

I am a server at Olive Garden, and one of the great advantages of working as a server is that it allows you a very flexible schedule. Being a busy BYU student I often find myself looking for servers to “pick-up” my shifts. Olive Garden employs a website called “Dish” that is used for a variety of tasks such as posting messages from the managers, employee purchases and schedules. However, the schedules site is something of a mess and doesn’t give me the immediate info I need in order to find someone to work for me. For example, there are roughly 30 different job classes and about 300 employees, but only certain employees can work certain jobs (e.g. server, bartender, grill, prep, etc.).

Overview of the System

What I did was create a VBA program that will work for any user with an Olive Garden “Dish” login. The program takes the user’s login credentials and first downloads and displays that user’s schedule. Following that, the program looks at that user’s job codes that are displayed in their weekly schedule and finds those correlated job schedules. For examples, if I am a server and a line cook, the program sees that I am scheduled as a server and a line cook and then goes and downloads the server and line cook schedules from “Dish”. Those schedules are then organized and displayed in their own worksheets.

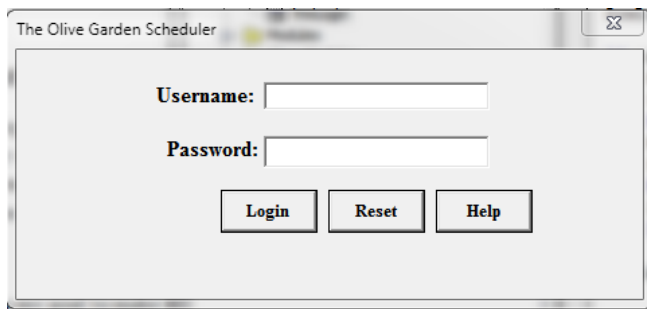
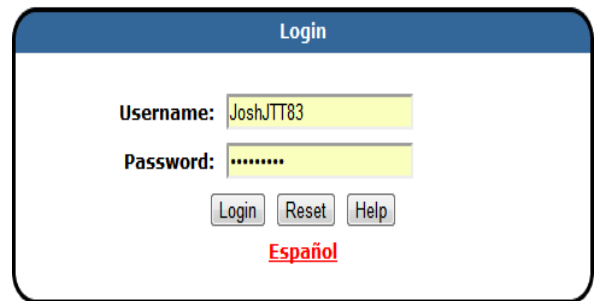
The user is then left with their job class schedules available and their weekly schedule displayed. The user can then click a button that will search the related job code schedules and will find every person that is not scheduled to work that job code that day. Under each day that the user is scheduled, there is a list of people who are available to work that job code and are available to work that day.

Implementation Documentation

In order to provide a concise, well-organized documentation of what I actually did for my solution, I will detail my program by describing the different steps (forms, sub-procedures, etc.) as they are executed by the user. In each section, I will describe the elements, why they were included, and how it is intended to be used in the task.

Login Form

When designing this program, it was my goal to make the different forms and pages look and feel as much like “Dish” as possible. So the initial form has the same text box titles and the same button titles as the login to “Dish”.

A screenshot of a web browser window titled "The Olive Garden Scheduler". The login form has a light gray background. It contains two text input fields: "Username:" and "Password:". Below the fields are three buttons: "Login", "Reset", and "Help".A screenshot of a "Login" form with a blue header bar. The "Username:" field is filled with "JoshJTT83" and the "Password:" field is filled with ".....". Below the fields are three buttons: "Login", "Reset", and "Help". At the bottom, there is a red link labeled "Español".

The user enters their credentials and can then select from three buttons. If they click on “Help” internet explorer opens to the related online help page. If the user selects “Reset” the text boxes are cleared. If the user selects “Login” and the username or password fields are empty, they are given a message box asking for the related details. After the user has entered their credentials and they click login the rest of the sub-procedures begin.

Login Procedure part 1

The first step of the login procedure inserts a large rectangle that informs the user that that program is connecting to the internet and to please wait until it is done. I did this because “Dish” is often a slow website (another advantage of having my schedule available on my computer!) that can take a long time to load. I then used the agent1 program to connect to the pages I needed.

The first problem I ran into was if the user was already logged into “Dish”, the agent1 would look for a username and password form that didn’t exist. To solve this problem, I wrote an if then statement that would only enter a username and password if there was a password to move to on the page.

```

agent1.visible = False
agent1.openpage "https://mydish.olivegarden.com"

If agent1.moveTo("password") Then
    agent1.explorer.document.all("user").Value = frmLogin.txtUsername.Text
    agent1.explorer.document.all("password").Value = frmLogin.txtPassword.Text
    agent1.explorer.document.all("submit1").Click
    agent1.waitForLoad
End If

agent1.openpage "https://pconweb.darden.com/applications/schedules/schedule.asp?rst=true"

```

Once the program has logged in, it would first navigate to that user's schedule and then import the page.

Build my Schedule Procedure

When I started designing this program, I actually first started by designing the schedule page to look and feel like the "Dish" page. In the back of my mind, I was thinking that I would need to later write the agent1 program to search, move to, and get text for each of the different aspects of my schedule from the source code.

Later, when I was working on the agent1 code, I realized that there was an import page command in the agent1 program, which made my job a lot easier! It was then that I realized that I didn't need to write a new page that would display my schedule, all I needed was to format the page that agent1 had imported. However, since I had already written the schedule page, I simply searched for the necessary data from the imported page and deleted that page after the data was parsed to My Schedule.

Beyond the formatting, the build my schedule sub-procedure deletes and creates any necessary sheets. It also inserts a "login box" that reloads the login form when clicked. There's also inserted a "find box" that will be explained later.

	A	B	C	D	E	F	G	H
1	Click here to login into Dish				Click here to find someone to work for me			
2								
3								
4	Schedule for Joshua Tolman							
5	From 12/6/2010 To 12/12/2010							
6	This schedule is current as of 12/9/2010 11:03:42 PM							
7	Team Member	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
8	Joshua Tolman	6:00:00 PM	5:30:00 PM	Day Off	5:15:00 PM	3:30:00 PM	Day Off	Day Off
9		DBD	DCL		DCL	DBD		
10		Server	Server		Server	Bartender		

Login Procedure part 2

Once the user's schedule is parsed, login then searches the job under each assigned shift and builds an array. In order to select that job's schedule from the drop list on the "Dish" website, I needed that job's source value. Agent1 was able to find that value in the source and then use that value to select that job and download and import that job's schedule.

Once that job schedule was imported, I wrote code that would find that job name and rename the import page to its related job, rather than its job code.

```
agent1.explorer.document.body.all("Schedtype")(1).Click

Dim x As Integer
Dim temp As String
Dim jobcodes As Variant
Dim jcode As String
Dim jobName As String

For x = 2 To 8
    If InStr(temp, Sheets("my schedule").Cells(10, x).Value) = 0 Then
        temp = temp & "|" & Sheets("my schedule").Cells(10, x).Value
    End If
Next

temp = Mid(temp, 2)
jobcodes = Split(temp, "|")

For x = 0 To UBound(jobcodes)
    agent1.position = 1
    agent1.moveTo jobcodes(x) & "</OPTION>"
    agent1.moveBackTo "value="
    jcode = agent1.getText(">")
    If jcode = 1005 Then jcode = 105

    agent1.explorer.document.all("jobName").Value = jcode
    agent1.explorer.document.all("submitForm").Click
    agent1.waitForLoad

    agent1.importPage jcode

    agent1.position = 1
    agent1.moveTo jcode & ">"
    jobName = agent1.getText("<")
Next

On Error Resume Next
Application.DisplayAlerts = False
Sheets(jobName & "'s Schedule").Delete
```

One problem I ran into here, is that there is a job called "Cafe Server" and every time agent 1 looked for "Server" it stopped at "Cafe Server." I thought of writing code that would search for

the next iteration of “Server” but since I know that management has recently stopped scheduling a café server and that they plan on deleting that job code, I felt the easiest answer was to write a simply if then statement. No other job names had this same problem.

Format Job Name Procedure

Having learned from my previous mistake, this procedure was much quicker to write than the My Schedule procedure. Each page that was imported was formatted to look like the “Dish” page. Beyond formatting, this procedure finds and delete’s rows without any data on them and delete’s rows that I didn’t feel were necessary.

	A	B	C	D	E	F	G	H
1	Schedule for all Server							
2	From 12/6/2010 To 12/12/2010							
3	This schedule is current as of 12/9/2010 11:03:46 PM							
4	Team Member	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
5	Lourdes Baquero	4:00:00 PM	4:15:00 PM	11:00:00 AM	j/c not scheduled	11:45:00 AM	11:15:00 AM	j/c not scheduled
6		DBD	DBD	LBD		LBD	LCL	
7	Jeremy Barney	5:15:00 PM	j/c not scheduled	11:45:00 AM	11:45:00 AM	5:00:00 PM	3:00:00 PM	j/c not scheduled
8		DBD		LCL	LBD	DCL	DCL	
9	Zach Boswell	6:00:00 PM	j/c not scheduled	j/c not scheduled	5:00:00 PM	5:15:00 PM	j/c not scheduled	j/c not scheduled
10		DBD			DBD	DCL		
11	Melissa Brinkerhoff	11:30:00 AM	j/c not scheduled	5:15:00 PM	11:45:00 AM	4:30:00 PM	3:00:00 PM	2:00:00 PM
12		LBD		DBD	LBD	DBD	DCL	DBD
13		4:45:00 PM						
14		DBD						
15	Brittany Brown	j/c not scheduled	4:15:00 PM	j/c not scheduled	11:45:00 AM	j/c not scheduled	12:00:00 PM	3:45:00 PM

Find People Procedure

The final aspect of this program is to provide a list of people who are available to work on the days that I am working and can work that job. The program searches every day that I work and finds the names of people who are not working that day but are still people on that job’s schedule.

```

Sub findPeople()

    Dim cx As Integer
    Dim rx As Integer
    Dim lx As Integer

    For cx = 2 To 8

        Sheets("My Schedule").Select

        lx = 12

        If Not Cells(8, cx).Value = "Day Off" Then

            Sheets(Cells(10, cx).Text & "'s Schedule").Select

            For rx = 5 To Range("A3000").End(xlUp).Row
                If Cells(rx, cx).Text = "j/c not scheduled" Then
                    Sheets("My Schedule").Cells(lx, cx).Value = ActiveSheet.Cells(rx, cx).End(xlToLeft).Text
                    lx = lx + 1
                End If
            Next
        End If
    Next

    Range("B12:H12").EntireColumn.AutoFit

End Sub

```

[Click here to login into Dish](#)

[Click here to find someone to work for me](#)

Schedule for Joshua Tolman

From 12/6/2010 To 12/12/2010

This schedule is current as of 12/9/2010 11:03:42 PM

Team Member	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Joshua Tolman	6:00:00 PM	5:30:00 PM	Day Off	5:15:00 PM	3:30:00 PM	Day Off	Day Off
	DBD	DCL		DCL	DBD		
	Server	Server		Server	Bartender		
	Brittany Brown	Jeremy Barney		Lourdes Baquero	Tyrell Motsinger		
	Ion Ciubotaru	Zach Boswell		Daniel Caine	Jason Smith		
	Zachary Elison	Melissa Brinkerhoff		Kinsey Collier			
	Marcos Gomez	Daniel Caine		Kate Dunaway			
	Brenda Gonzalez	Kinsey Collier		Carla Garcia			
	Julia Gould	Kate Dunaway		Justin Hurlburt			
	Gary Greene	Lucy Dunn		Michael Jacob			

Learning and Conceptual Difficulties Encountered

There were several other things that I had hoped to accomplish but was unable too. I very much look forward to fixing those things over the break. There is a problem in the form sometimes crashing excel, but Prof. Allen told me not to worry about that and to see him after finals to get it fixed. The search is somewhat incomplete. It only finds people who are not scheduled to work that *day*. What I want it to do is to find people who are not scheduled to work that *shift* (e.g. a server that is schedule to work that day's lunch shift will not show up on my list, even though they may be available to work my night shift).

Regarding the search for people to work, I had planned on writing code that would eliminate people from the list if they were scheduled for 40 hours. I may still try and add this in later, but with the amount of shifts being moved around it is often the case that someone may be scheduled 40 hours but not have worked 40 hours. Further, there are very few people who get over 20 hours a week; much less 40 (most people there are students or work multiple jobs). I think in the future I would simply write code that highlights the names of people scheduled 40 hours.

Lastly, I plan on writing code that will also list people that I could "pick-up" for on days that I'm *not* scheduled. This was more of a time constraint; I look forward to adding this in later as well.

I was impressed by how much I needed to conceptually plan and map out my project. Having done it, I have a huge mental list of things I would do differently next time and plan on fixing over the break. When I started, I had a general idea of what I wanted and what I wanted it to look like, but how to get there was a mystery to me. I've learned that next time I will try and write out the different steps and rough ideas for procedures and their purposes. I think this will help me to better understand what is necessary from the beginning. Rather than writing code and finding out later I didn't really need it.

I learned a lot about how computer's think and work. I remember when the code for the search worked on the first day of the week, but not on the second. I had to think about each step of what the computer was doing and where it was before I figured out that it was ending on the wrong worksheet to continue.

I loved this project. I can't think of any other project I've done here at BYU in which I have learned more. I feel that the skills taught in this class are only understood once one tries to actually use them and with this project I finally began to understand VBA.