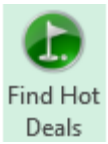
Hot Deals Finder

Executive Summary.

Avid golfers are always looking for good deals for rounds of golf. Despite how much we love golf, we do not love the high price tag that often accompanies our hobby. My goal with this project was to create a way for golfers to get daily emails about great deals on local golf courses, without any searching on their part. The system that I built will go online and find good deals on local golf courses, based on your current position, in order to notify the player via email if there are any good local deals that the player should be aware of. I believe that busy professionals who are avid golfers would be willing to pay a price in order to receive automatic daily notifications about great golf deals in their area.

Implementation Documentation.

1. Finding Hot Deals 

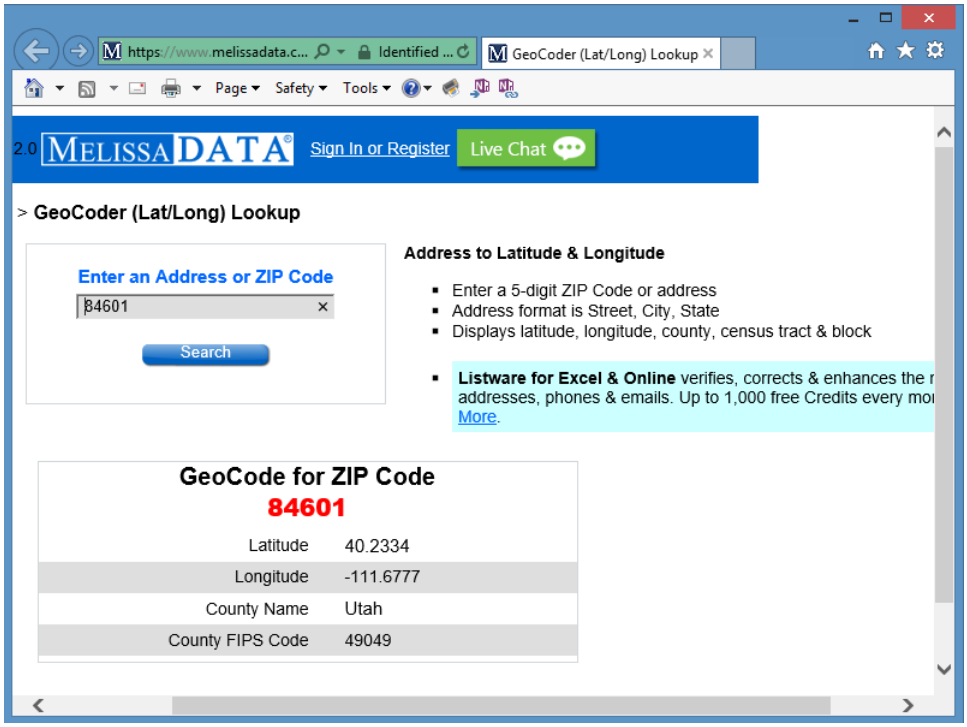
As stated earlier, golfers love to find good deals. Fortunately, a web site called Golfnow.com has been built in recent years that works with golf courses to find slow times during the day and posts these tee times on their site at discounted prices. Discounts can be well over 50% off, depending on the golf course and the time of day. These deeply discounted tee times are called “hot deals” on the website. This program will search golfnow.com for these hot deals on a day and location (zip code) specified in the worksheet. It will then take the course names, prices, and tee times and put them into the output cells in the “HotDeals” worksheet.

Golfnow.com uses latitude and longitude rather than addresses or zip codes to search areas for golf deals. Because I don't expect users to know the exact latitude and longitude of the area they want to play golf in, I have built an input section in the worksheet where the user can input the day and zip code of when and where they want to play.

Zip Code	Date
84601	Thu+Apr+7+2016

The Date is formatted with plus signs in between each of the parts because the website requires it to be in this format in order to search for the correct day. However, the user can input the date in most formats and the cell will recognize it and change it.

The first part of the program takes the zip code given by the user and uses it as input into a website called Melissadata.com that finds latitude and longitude based on a zip code given.



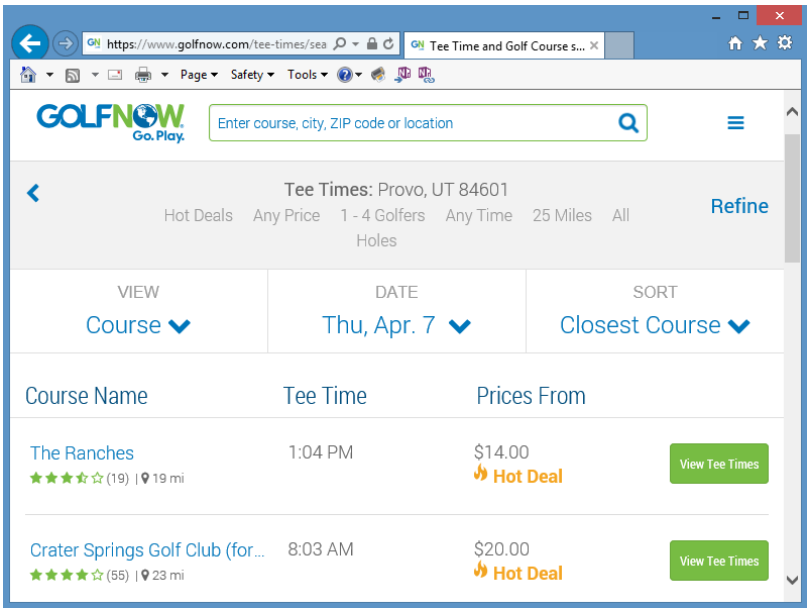
As can be seen in the above screenshot, this website takes the address or zip code given and finds the latitude and longitude. The program then searches through the HTML code on the webpage and finds the latitude and longitude and saves them into variables.

The next section in the program takes the latitude and longitude found on the previous website along with the date provided by the user and plugs them into a golfnow.com search URL template that looks like this:

"https://www.golfnow.com/tee-times/search#radius=25&latitude=" & Latitude & "&longitude=" & Longitude & "&sortby=Facilities.Distance.0&date=" & Worksheets("HotDeals").Cells(2, 2).Value & "&hotdealsonly=true&timemax=48&view=Course"

Latitude and Longitude are the names of the variables that are placed in the URL outside of quotation marks and cells(2,2) refers to the date from the worksheet.

Next, using this URL, the program opens the webpage that specifies the search terms. The website displays the results as follows:



For this particular search in Provo, UT on April 7, there were only two golf courses found. We have seen as many as thirty displayed on the page, though.

Next, the program searches through the HTML code of the search results page and find the course name, tee time, and price. These are then stored in separate variables, after which they are placed into the output cells in worksheet.

Golf Course	Price	Tee Times
The Ranches	$14.00	1:04 PM
Crater Springs Golf Club (formally Homestead)	$20.00	8:03 AM

As can be seen, the output in the excel worksheet is the exact data from the website above. Before placing the information into the output cells, the program also cleans any extra white space from the text so that the output and email will both have professional looking formatting.

If there are no hot deals available for the area searched, the program will insert the following into the output cells:

Golf Course	Price	Tee Times
There are no Hot Deals for the time and place specified		

2. Sending the email

Once the data from the website has been placed in the output cells in the worksheet, the user can then have the program send them an email with the information contained in the

output cells in it. The email is written from a basic template stored in a .txt file that reads as follows:

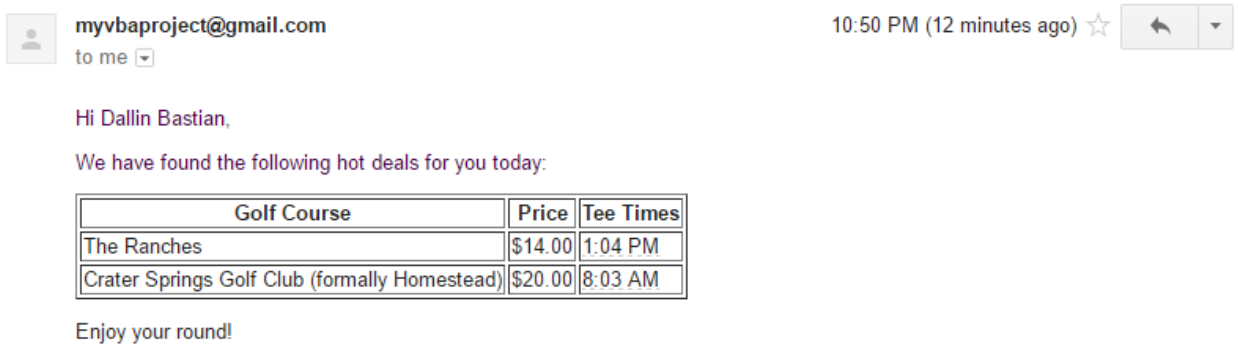
```
Your Hot Deals for the day
<HTML><head><title>message</title></head><body>
<p>Hi <NAME>,</p>
<p>We have found the following hot deals for you today:</p>

<COURSEDATA>

<p>Enjoy your round!</p>
</body>
</html>
```

The program will look at the first line in this text and use that as the email subject. The rest of this message is written in HTML code which allows us to put the data into an easy-to-read table in the email. Every time an email is sent, the program will look at the <NAME> tag and replace it with the username of the person whose computer is using the program. It will then look at the <COURSEDATA> tag and replace it with the names of the courses, prices, and tee times that were found using the Hot Deals Finder in the previous section.

The program uses a function that connects to the Gmail smtp mail server. It uses a made up username and password that I created for this project. The email address of this account that all the hot deals emails will get sent from is myvbaproject@gmail.com. Using the made up login credentials, the function is able to connect to the Gmail server and send an email of the populated template from the .txt file. Because the function that sends the email is a Boolean function, I needed to specify what would happened if it is true (the email is successfully sent) or false (the email fails to send). So, if the function is true, it will place the time the email was sent in the output cells in the worksheet.

Golf Course	Price	Tee Times	Email Sent?
The Ranches	$14.00	1:04 PM	4/6/2016 22:50
Crater Springs Golf Club (formally Homestead)	$20.00	8:03 AM	

This is the same information that was found earlier. The program successfully sent an email on April 6, 2016. When the email comes through, it looks like this:



myvbaproject@gmail.com 10:50 PM (12 minutes ago)
to me

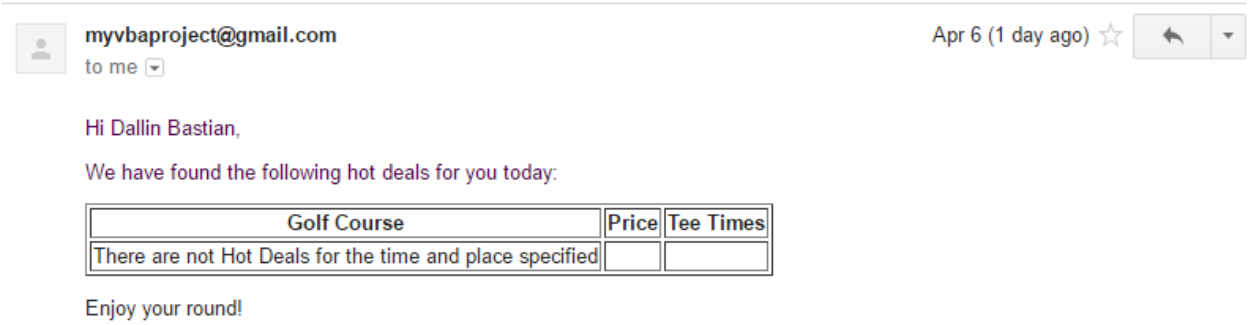
Hi Dallin Bastian,

We have found the following hot deals for you today:

Golf Course	Price	Tee Times
The Ranches	$14.00	1:04 PM
Crater Springs Golf Club (formally Homestead)	$20.00	8:03 AM

Enjoy your round!

If there are not hot deals available, the email will look like this:



myvbaproject@gmail.com Apr 6 (1 day ago)
to me

Hi Dallin Bastian,

We have found the following hot deals for you today:

Golf Course	Price	Tee Times
There are not Hot Deals for the time and place specified		

Enjoy your round!

If for some reason the email fails to send, it will put “Failed” in the output cells and the program will print the error description in the immediate window in the visual basic editor to give the user more information on why the email is not sending.

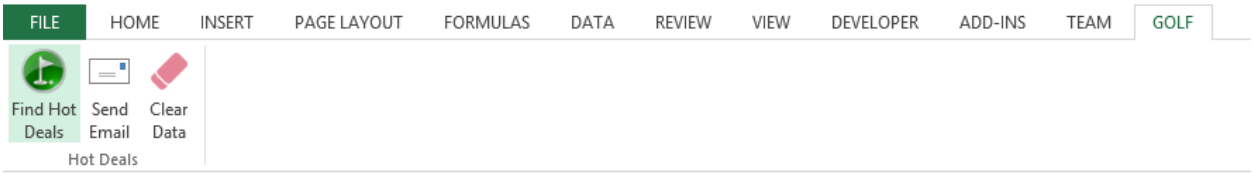
Golf Course	Price	Tee Times	Email Sent?
The Ranches	$14.00	1:04 PM	Failed
Crater Springs Golf Club (formally Homestead)	$20.00	8:03 AM	

Finally, I have put functionality into the program to also send emails as text rather than HTML. However, the HTML has a much better format and makes the emails look much better.

3. Deleting existing data 

Finally, there is a short macro that will clear any data that currently resides in the output cells so there won't be any issues with overlapping data from prior searches.

For this project I have created a new ribbon in excel called "Ggolf" that includes the three buttons I have just discussed. The entire ribbon looks like this:



I will most likely combine these three buttons into one for practical every-day use of this program so users can have their computer automatically run one program once or twice a day and get the email, which was the goal with this project. However, for illustration purposes, I separated them into three buttons to clearly show that there are three sub-procedures that do different things.

Discussion of learning and conceptual difficulties encountered.

Working with difficult websites.

Before this project, I had only used VBA code with straightforward, easy to use websites. I had a lot of frustration early on in this project because the website seemed to do something

different every time I tried to run the program. It would be working fine one night when I went to bed, and I'd wake up the next morning and it wouldn't work at all. One of the biggest problems was trying to get Golfnow.com to use the template URL that I discussed earlier in this document. Most of the time, it would just take me to deals for Orlando, Florida and the current date despite what I put into the URL, and occasionally actually take me to the search page I wanted it to. Eventually, I discovered that the website didn't like to be told where to go when initially visiting the website, but would work if I was already on the website and then put in the URL template. So, I have the program load the page twice- once to get it to golfnow.com, and a second time to actually load the results I wanted.

Also, because of how Golfnow.com is set up, you cannot find the results of your search by merely viewing the source of the webpage. I spent a lot of time being frustrated before finally learning that I needed to inspect the element of the current page in order to find the data I was looking for.

Formatting text.

This project really taught me how frustrating formatting text can be. I finally got this program to where it was bringing in the correct information, but it would bring it in a really strange formats, sometimes with many lines of white space before and after the text I actually needed. I first tried to use the "trim" function to trim all of the spaces from the text, and this helped sometimes, but I was still getting significant areas of white space that apparently were not spaces. Trim only works with spaces. I eventually used the ASC function, which returns a numeric value for a character in a string of text. It turns out that any character with an ASC

value of less than 32 is white space, so I now have the program run through a loop with each piece of text returned from Golfnow.com and evaluate each character in the string. If a character returns an ASC value of less than 32, then the program turns that character into a space. After the program has gone through each of the characters in the string, it then trims the string, eliminating all the spaces and leaving only the text desired. Now the information in the output cells and the email is consistently formatted and looks much more professional than before.

Assistance.

I worked with Dr. Gove Allen on a few different issues. First, he helped me work through why the webpage was sometimes visiting the correct URL template and other times just going to Orlando, Florida. He thought it might help to load the webpage twice, and that has fixed all the issues I had with the webpage not going to the correct search results. He also showed me how to find the correct search data by inspecting the element of the webpage rather than viewing the source.

Finally, he assisted me with formatting. He showed me how the ASC function worked, which I had not known about before, and I was able to successfully implement it to trim the whitespace from my data. He also wrote the HTML code that the program uses to create the table to place the data in when sending an email, which makes the email appear much more professional.