Patrick Nelson
VBA Final Project

EXECUTIVE SUMMARY

Flight delays are frustrating for everyone affected by them. The passengers of the flight; the families, friends and employers of the passengers who have made arrangements either in the flight's origin or destination; and the delayed airlines. No matter what causes the delay, the passengers will be upset. And when passengers are upset, the reputation of the airline with the delay will suffer the consequences. If the delays happen frequently enough, passengers may stop using that particular airline and start using a competitor. And since we don't live in a perfect world, these flight delays will continue to happen even with the best managed airlines.

This Final Project does not try to eliminate flight delays or run airlines more efficiently. The objective of this project is to mitigate some of the pain associated with the inevitable flight delays. As email- and text message-enabled mobile devices are becoming ubiquitous, I have written VBA Code that retrieves online flight status information and sends email and text message status updates to passengers.

Many airlines send an email (or several emails) to their customers before a flight takes off. These emails confirm itineraries and announce when passengers can check in to the flights. My VBA code sends an email and a text message to passengers announcing the status of the flight several hours before it is scheduled to depart. Whether the flight is on time or delayed, passengers will receive the appropriate status update. If the flight is on-time, the passengers will be pleased. If the flight is delayed, the passengers may not be happy, but at least will be able to plan their trip better.

Business Application

This VBA code can be used by airlines to stay in touch with their customers and provide them with real-time flight status information. Airlines are constantly promoting how customer-friendly they are. Airlines market their flights as being clean, comfortable, and on time. They also market the on-flight perks. After all, commercial airlines are in the customer service business. But airlines have a long way to go regarding the quality of their customer service. This VBA code can be considered one small step to provide a higher level of customer service to its passengers.

The VBA code is pretty raw at this point. To make it work more completely, airlines would have to buy-in to the idea of emailing/texting flight status updates to its customers. For example, an airline would have to give me (the author of the code) access to passenger information, including names, contact information, contact preferences and flight status information. With access to this information, I could revise my VBA code in order to fully automate the process of sending flight status updates to airline passengers. However, since I don't have access to this information I have created my own passenger list, which simulates data that would be imported from an online source. I believe that attempting to "hack" into airline databases to find passenger information would be out of the scope of this MBA course (legal and ethical concerns, aside).

IMPLEMENTATION DOCUMENTATION

To create this VBA code, first I wrote code that enabled VBA to retrieve password-protected flight status data from an online source. The data includes passenger names, contact information, flight information, and flight status (i.e., delays) information. In order to get access to this information, I would have to make a deal with the airlines and sell them on my idea. Assuming I had such access, my VBA codes would import this data into an Excel spreadsheet (into the "passengers" tab). However, since I do not have access to this flight status information, I simply created the information in the spreadsheet *as if* it had been imported from the password-protected data source.

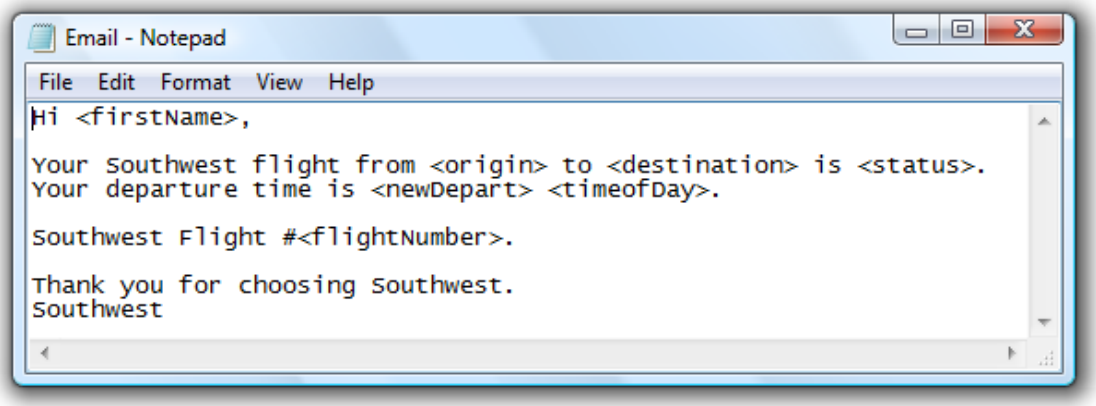
Once the passenger and flight status data is imported into the "passengers" tab, my VBA code is ready to interpret and communicate with the passengers. The next piece of code that I wrote is a *function* that returns the contents of text files. The text files that I need to access are the email and text message templates that my VBA code modifies and sends out to passengers. These templates are saved in Notepad "text documents" and the function is shown below.

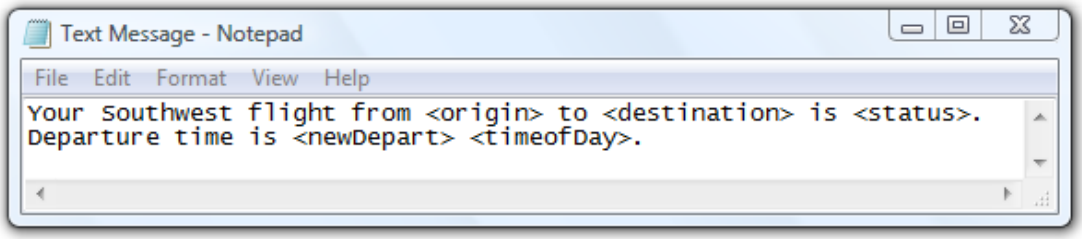
```
Function readFile(path As String) As String
  ' returns the contents of a text file
  Open path For Input As #1
    readFile = Input(LOF(1), #1)
  Close #1
End Function
```

These two lines of code show how I labeled the email and text message templates. Using these labels made it much easier for me to alter the content of the templates.

```
emailTemplate = readFile(ThisWorkbook.path & "\Email.txt")
textMessageTemplate = readFile(ThisWorkbook.path & "\Text Message.txt")
```

Here are the email and text message templates used to send to passengers. My VBA code will insert the appropriate customer information (i.e., name and flight status information, etc.) in place of the information inside the "< >". For example, each passenger's first name will be inserted into <firstname>.





The following is the part of my VBA code that inserts the passengers' name and flight information into the Text Document templates. The Code then sends an email message to each passenger who has indicated he wants to receive one. For example, if the passenger does not wish to receive an email message, he can indicate this preference when he purchases his plane ticket. This communication preference will then appear as "Y" for yes or "N" for no in the Excel spreadsheet with the rest of his contact information. A similar code is used to send text messages.

```
row = 2
Do Until Sheets("passengers").Cells(row, 2).Value = ""
    If Sheets("passengers").Cells(row, 5).Value = "N" Then row = row + 1
    End If
  email = Replace(emailTemplate, "<firstName>", WorksheetFunction.Proper(Sheets("passengers").Cells(row, 2).Value))
  email = Replace(email, "<origin>", Sheets("passengers").Cells(row, 8).Value)
  email = Replace(email, "<destination>", Sheets("passengers").Cells(row, 9).Value)
  email = Replace(email, "<flightNumber>", Sheets("passengers").Cells(row, 10).Value)
  email = Replace(email, "<status>", Sheets("passengers").Cells(row, 12).Value)
  email = Replace(email, "<newDepart>", Sheets("passengers").Cells(row, 14).Value)
  email = Replace(email, "<timeofDay>", Sheets("passengers").Cells(row, 15).Value)


  Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 4).Value, _
                                    frmPassword.txtUsername, frmPassword.txtPassword, _
                                    "Flight Status", email)

  row = row + 1
Loop
```

Notice that the following piece of the VBA code only sends an email or text message to passengers who want to receive such a message. The spreadsheet indicates whether each passenger prefers email and/or text message communication by marking either a "Y" or "N" in the 5th column. If and when a passenger does not want to receive an email or text message, an "N" is marked in the appropriate column. The VBA code will then move on to the next passenger in the spreadsheet.

```
If Sheets("passengers").Cells(row, 5).Value = "N" Then row = row + 1
End If
```

The following code enables the VBA code to send flight status updates via all the major cellular phone service providers: Verizon, AT&T, Sprint, T-Mobile, US Cellular, Virgin Mobile. Depending on which cellular service provide the passenger uses, the code will use the appropriate notation to send the text message (i.e., "@vtext.com" for Verizon).

```
row = 2
Do Until Sheets("passengers").Cells(row, 2).Value = ""
  textMessage = email

    If Sheets("passengers").Cells(row, 7).Value = "N" Then row = row + 1
    End If

    If Sheets("passengers").Cells(row, 8).Value = "Verizon" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@txt.att.net", _
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

    If Sheets("passengers").Cells(row, 8).Value = "ATT" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@vtext.com", _
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

    If Sheets("passengers").Cells(row, 8).Value = "Tmobile" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@tmomail.net", _
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

    If Sheets("passengers").Cells(row, 8).Value = "Sprint" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@messaging.sprintpcs.com",
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

    If Sheets("passengers").Cells(row, 8).Value = "Virgin" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@vmobl.com", _
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

    If Sheets("passengers").Cells(row, 8).Value = "US Cellular" Then
    Sheets("passengers").Cells(row, 1).Value = sendGMail(Sheets("passengers").Cells(row, 6).Value & "@email.uscc.net", _
                                               frmPassword.txtUsername, frmPassword.txtPassword, "", textMessage)
    End If

  row = row + 1
Loop
```

Since airlines update flight statuses very frequently, I also had to write code that would give passengers these frequent updates. My code moves the data on the "passengers" spreadsheet tab to the "old" tab and imports the updated flight status data into the "passengers" tab from the online data source. I also wrote a separate sub procedure ("ifDifferent") that compares the data in the two spreadsheet tabs. If there are differences- either new passengers or flight status information- the VBA

code will run its original task (sub procedure "flightStatus") of communicating via email or text message the flight status information to the passengers with the updated information. After this process takes place, my code then deletes the data in the "old" tab. This process repeats every 15 minutes.

Even though the VBA code automatically runs every 15 minutes, passengers certainly would not want to receive a status update every time the code runs. Consequently, after the new flight status information is imported from the online data source, my code checks for differences in passenger and flight information. When the code detects differences in information- either in passenger or flight status- an email and/or text message are sent to the passengers with the flight status information. Therefore, as new passengers are added to the flight and as delays occur, affected passengers receive the relevant status update (i.e., new passengers receive a status alert and existing passengers receive a status update, if applicable).

CHALLENGES ENCOUNTERED and LESSONS LEARNED

This course has been my first encounter with VBA programming and consequently, I came across many challenges while writing the code for this final project. As writing VBA code does not come naturally to me, I had to ask several questions along the way to others with more experience with VBA. However, during this process I learned a great deal about VBA programming.

The first notable challenge I encountered was that each passenger listed on the "passengers" tab would receive an email or text message with his/her flight status update every time the code runs. To remedy this problem, I added another line of code so that passengers only get an email and/or test message when they are new to the list ("passengers" tab) or when their flight status changes. I did this by telling the code to skip to the next row (passenger) if and when the passenger information was the same as before (i.e., passenger and flight status were the same as the last time the code ran). Although looking back, this challenge seems very elementary to me. However, at the time the challenge made me think quite a bit.

Another lesson I learned while completing this project was how to run my VBA code automatically without clicking the "Play" button. I did on online search to find the appropriate code to do this. I used the following code to run my program every 15 minutes:

```
Public RunWhen As Double
Public Const cRunIntervalSeconds = 900 ' 15 minutes
Public Const cRunWhat = "ifDifferent"
```

Notice that every 15 minutes, my code is programmed to run the “ifDifferent” sub procedure. The “ifDifferent” sub procedure compares the two spreadsheet tabs (“passengers” and “old”) to check if there are any new passengers or flight status information. If and when there are differences, the “ifDifferent” sub procedure prompts the primary sub procedure of my VBA code, “flightStatus”, which sends emails and text messages to passengers.

A third lesson I learned while completing this final project was how to move data from one spreadsheet tab to another. Again, this now seems to be a very simple ability in VBA. However, at the time I did not know how to do this. I initially got the idea to use two tabs from a friend who took the VBA course last year. When I told him about my project, he asked me how I would send out flight status information to passengers when their respectively flight status changed. After thinking for some time, I got the idea to create two tabs: an updated tab (“passengers”) and a tab with older information (“old”). After I created another tab, I wrote VBA code that compares the data in the two tabs, cell by cell. When there are differences in the data (i.e., new passenger, contact information, or flight status, etc.), the code sends that passenger an email and/or text message.

The challenge to this is that the passengers must be listed in the exact same order each time the data imports from the online data source. Otherwise, the code would detect differences and email and text message every passenger on the list *even if their individual information had not changed.* I still have not figured out how to change my code so the order in which the data is imported does not affect which passengers receive a flight status update via email or text message.

Overall, I learned a great deal while completing this final project. And although I am not even close to being a VBA expert, I have learned many of the basis of VBA programming that should help advance my professional abilities.