README

Author: Jameson Thai

This document is a general guide to setting up or maintaining the Ticketing System V 2.2. Long title right? Well this document is going to be longer, so grab some popcorn and watch the show.

Setup

When making a copy of the original ticketing system, there are a couple of steps you need to modify and implement before programming!

1. Firstly, you need to set the ACTIVE TRIGGERS for both the spreadsheet and the form. To access these triggers, you need to go into the SCRIPT EDITOR!
   Spreadsheet: Tools -> SCRIPT EDITOR
   Form: 3 dots on the top right hand side -> script editor
   a. From the script editors, you go into EDIT in the toolbar and drop down to “All Your Triggers” .
   b. From there, you set the trigger: onEdit, From Spreadsheet, and onEdit for the spreadsheet. For the Form: onFormSubmit, From form, and onFormSubmit.
   c. Note: For the triggers, make sure there’s only one person who set up the triggers because it will either duplicate the processes or make your execution time very slow.
2. The next thing you need to set up is the links to the Master sheet, and respective category sheets like Web, Database, Editorial, Media, and Graphic Design.

Testing

1. You test the entire system by using the form and submit a query which updates into the spreadsheet. From the spreadsheet, we can observe the changes and it also filters out into the respective sheet.
2. Another way to test is to use the console log, for mac command + enter. In the statement you want to test you just type in Logger.log(object) and it will function like a system.print or a print
3. Remember to always check the execution transcript

SpreadSheet

Methods:

onOpen:

- This function operates every time the spreadsheet is opened, the main purpose for this function at the moment is to calculate how many days until the deadline and put in the appropriate response.
- The responses are #Days till deadline, Deadline is today, or Past Deadline!
- Uses getOtherSheet to grab the corresponding sheet to update to.
- This method requires a lot of execution time and is very inefficient so a future update is to improve the efficiency of this execution.

getOtherSheet:

- This method determines what sheet to use and returns their websheet ID.

onEdit:

- This method includes changedAssignedCellBackground, updateCorrespondingSheet, assignedEmailStatusUpdate, darkenResolvedTickets and resetStatusColor.

changeAssignedCellBackground:

- This method

updateCorrespondingSheet:

- This method updates the corresponding sheet depending on which sheet was the active sheet. It allows for bidirectional updates WHERE if I update in master it will update in the appropriate category like web. The same logic applies the other way around where if I update in the web sheet, it will also update in web. This method only extracts the information and gets it ready to be executed in setTixInformation.

setTixInformation:

- This method is the true method that actually allows for the bidirectional functionality. Essentially it sets the status, person and coloring in the other sheet that also needs to be updated.

assignedEmailStatusUpdate:

- This method emails the client and the worker who assigned the task to themselves. It will send the client a ticket receipt with a message from the College of H&A that says who took on the task and their email. From the College's perspective, we get an email saying we got assigned to the task.

darkenResolvedTickets:

- This method basically darkens resolved rows and sends an email to the client saying that the class has been resolved. This is the last step in the onedit() method.

resetStatusColor:

- This method essentially clears the color if it was assigned a new tag in the status column. This is an obsolete method that can be aggregated into a singular method. Going to be depreciated

getEmployeeEmail:

- This method gets the employee by email to pass into the assigned email updates to send emails to the respective employee who took on the task.

getColIndexByName:

- This method has parameters that obtains the desired column name to get the index of within that spreadsheet. It returns the index of the desired column within the sheet.

Form

doGet:

- This method just obtains the submission from users of the google forms

onFormSubmit:

- This method is essentially what runs the program as it formulates the response to the object, makes the ticket document, populate the spreadsheet and send the email notification.

formResponseToObject:

- This creates a working object that could be passed between functions. Additionally calls the create the Ticket Document which this information is passed into.

createTicketDocument:

- This method takes in the object that has the submission information and formats it into a ticket.

populateSpreadsheet:

- This method populates not only the master sheet, but also the respective sheet determined by the ticket parameters. For example, if the ticket parameter for ‘Project Focus’ stated it was ‘Web’, it would populate the Web and Master sheet with the ticket information. Of course, not all information is inputted thus there is a category that directs employees to the ticket which has more detailed information.

sendEmailNotification:

- This method sends an email to the requester stating that the team has received their request and will notify them when someone takes the task. In this email, it also sends the ticket receipt for the client’s request.

addOpenByIDGooglePrefix:

- This method takes IDs from "Files to attatch" question on the form and adds the open by ID url prefix to each ticket. It helps the onFormSubmit() method.

createDuplicateDocument:

- This method creates a duplicate ticket and returns a new document with a given name from the original. It is used in createTicketDocument when creating the ticket ID shortlinks.

Important Notes:

1) If possible don't make a copy of the copy because it messes with the server reference… Took a long time to identify and solve the problem. Just move the testing document up a level in the directory hierarchy.
2) Make sure only one person has triggers set in the project
3) A lot of the code remains inefficient so a future task is to optimize it