**Project Title: Token System**

This project is about using Google sheets API to read a spreadsheet and take some actions based on the data.

Many of you may have experienced getting tokens (and time slots) to avoid unnecessary waiting times (e.g. passport office or doctor’s appointment). In this project, we will build a simple token system that will mimic a virtual queue and assign tokens to people based on the order in which people filled a form.

(NOTE: THIS MAY BE A CHALLENGING PROJECT, SINCE WE HAVE NOT COVERED THIS AS PART OF THE SYLLABUS. BUT NONETHELESS IT IS AN INTERESTING APPLICATION OF SPREADSHEETS. TAKE IT UP ONLY IF YOU WANT TO EXPLORE SOMETHING NEW. There is good amount of sample code in Internet to get started.)

**Basic Requirements:**

Setup a google form which among other things will positively collect name, email addresses, gender, location choices (say from among 5 choices). Ensure that the collected data is available as a spreadsheet. Or you create a spreadsheet yourself and ensure these columns are present.

The Google Sheets API is a RESTful interface that lets you read and modify a spreadsheet's data. You will now programmatically parse this spreadsheet data and send an email to the form-filler. The email will contain the token number along with appointment time.

1. Assume that the form is open at 8am everyday and closes by 8.30am (you don’t have to code this). Once the form closes, the spreadsheet is parsed. Assume the appointments start at 10am and go on till 1pm and each form-filler will get a 30 min appointment slot.
2. In the email you send the form-filler, address the user using the correct name as filled in the form (e.g. Dear Ravi). Then put some text, but be sure to include within the token number and appointment time (assume tokens start from 1 i.e. the first one to fill the form will get token 1 and appointment at 10am, second will get token 2 and appointment at 10.30 etc). And if there are more people than available slots for that day, the email to those whose appointment cannot be fulfilled, can be assigned a token as – and appointment time as “Not available”.
3. If someone fills the form more than once, token should not be wasted. Another email can still go, but the token number (and appointment time) should still be the one corresponding to when this person filled the form first.

**Customization:**

You can extend to add further features. Here are some examples, but really you should come up with your own. Marks will be given per YOUR creativity.

1. Maintain gender specific queues (ignore location here), wherein assume that each gender has a separate queue.
2. Maintain a location specific queue. For example, say 3 applied for location1 and 10 applied for location 2. Then the 3 who applied for location1 will get token 1, 2 and 3 (and corresponding appointment slots). And 6 out of the 10 will get appointments for location2 and rest no appointments.
3. Combain gender with location i.e. each location now has 2 queues, one for male and one for female.

**Marks Distribution (20 marks):** These marks will be awarded via a viva, where you demonstrate the project and TAs will ask questions. Note, for this project you can demo using your own Google account. But be sure to cover relevant test cases that showcase the functionality you coded.

* Email sent to the form-filler: 3 marks
* Email with correct token and appointment information: 5 marks
* Duplicate form-filling handling: 2 marks
* Customization: 4 marks
* Viva: 4 marks
* Code quality, organization, and comments, latex based report: 2 marks

Note: All submitted code in a given project will be checked for plagiarism and if caught, your case will be forwarded to DDAC.

**Upload:** Please upload all relevant files, including report as a zip file on BodhiTree before the deadline. NO CHANGES TO THE FILES WILL BE PERMITTED POST THIS, INCLUDING DURING EVALUATION. During evaluation, TAs will download the zip from Bodhitree, and ensure you are using this downloaded code in demonstration.

**References:**

<https://developers.google.com/sheets/api/guides/concepts>

<https://spreadsheet.dev/send-an-email-for-every-row-in-a-google-sheet>

<https://www.codementor.io/@olatundegaruba/google-apps-script-automated-emails-m2m0ojq9v>