CS104 Project
Spreadsheet Programming

Arnav Garg
22B1021

Contents

1 Introduction

This report aims to provide a comprehensive description of the CS104 project on Spreadsheet Programming.
Here we implement a software to read and edit Google Sheets API to create a static system to deal with responses received through Google Forms.

2 Project Overview

We here implement a system for booking appointments at a multi locational health centre named Hope clinic.
Form users have the option to fill in their time preference and their preferred location and they will be assigned a time slot based on their sex and chosen location.
It also implements an algorithm to give a preference of slots to the users who fill the form first.[1]

3 Basic Structure

There are 10 queues in total- 5x2 for location and sex. I have implemented a system to prevent duplicacy by assigning a token number of 0 which is indicative to my code to send a mail to the form filler informing them of the redundancy. The Google Sheets are read through a service account using the Google Sheets API[1] implementing the `gspread`[2] library in python. To execute the files, the user needs to run the `script.py` file after uploading relevant testcases through the google form. I then account for 7 slots-

1. 10AM-10:30AM
2. 10:30AM-11AM
3. 11AM-11:30AM
4. 11:30AM-12PM
5. 12PM-12:30PM
6. 12:30PM-1PM
7. 1PM-1:30PM

with each slot corresponding to one token number.
Now I establish a SMTP connection to the GMail host [3] and send out auto generated emails using inbuilt libraries which specify the details of the appointment assigned to the user with `html` formatting.

4 Customizations

4.1 Location specificity

My thought process for the project went as follows- when a person books an appointment at a chain of clinics widely distributed across the country they can choose a specific location for which they are available to attend as locations are pretty spread out. Thus we can safely maintain location specific queues with a person able to book an appointment for only one possible location.

[1] I also tried and played around with optimization of the assignment algorithm but was unable to implement it efficiently within constraints.

4.2 Gender queues

Implementing a sex based independent queue for both males and females to streamline the process of obtaining treatment is a natural consequence for distributed clinics addressing problems specifics to both the sexes. I have 2 queues at each of my 5 locations - making a total of 10 queues with each of them having a unique token number system starting from 1.
A token number of 0 indicates duplicacy in token filling.

4.3 Optimization algorithm and time preference

I have added an option for the form filler to choose multiple time slots out of the 7 given to indicate their availability and assist us in assigning a time slot favourable to them. However this poses a problem, while we can trivially assign a method for assignment, by just assigning the first or last selected slot, it raises problems of optimization. This can be shown by taking examples for as less as 3 users and 4 slots. This dips into the realm of **matching theory of bipartite graphs**[7] which turned out to be tough to implement. However an efficient algorithm was developed but not implemented to keep time and space complexities feasible for very large values.

Claim: An example of an efficient algorithm is assignment in **order for which slot has the least takers**, then **assignment by ascending order of token number** and then finally in a trivial order.

4.4 Email modification

I have modified the bodies through the `MIMEText` function of my email to parse `html`[4], so that my emails can display formatted text. I also added a check for age which prompts minors to be accompanied by a guardian and senior citizens to avail a special discount.

5 Important Links

- Google Form
- Google Sheets
- GitHub Repo

References

[1] URL: https://developers.google.com/sheets/api/guides/concepts.

[2] URL: https://medium.com/geekculture/2-easy-ways-to-read-google-sheets-data-using-python-9e7ef366c775#c6bb.

[3] URL: https://spreadsheet.dev/send-an-email-for-every-row-in-a-google-sheet.

[4] URL: https://stackoverflow.com/questions/882712/send-html-emails-with-python.

[5] URL: https://www.docs.gspread.org.

[6] URL: https://www.globalindian.com/givingback/british-indian-doctors-helps-indian-doctors/.

[7] Michael D. Plummer Laszlo Lovasz. *Matching Theory*. The American Mathematical Society.