COMP1002 group project report

Group 36

Member 1 Name: HU Wenqing	Member 1 Student ID: 21094549d
Member 2 Name: WEI Xianger	Member 2 Student ID: 21094754d
Member 3 Name: ZHOU Siyu	Member 3 Student ID: 21094655d
Member 4 Name: IEONG MEILENG	Member 4 Student ID: 21108573d

ii. Problem description

As the covid-19 pandemic background, the vaccination system is highly necessary for universities so that a safe teaching environment can be provided to all staff members and students. We have developed this system to record and track the number of staffs/students who have got vaccinated. This vaccination tracking system can be used by students, staff and the administrator. For students and staff, they can input their vaccination information into the system. For administrators, he/she can access the system to search and also change all of the vaccination information in the system. The new function we designed is the reminder email for the third vaccination. The Government has been encouraging the public to get the third shot of the vaccine after the sixth month of the second injection and the third injection will strengthen the body's immunity to the coronavirus. So we designed this function to remind the staffs/students who have had their second dose of vaccine over 180 days to take their third vaccination to prevent the virus.

iii. Data Abstraction

We have used several data types in our program. The first one is the numbers. We have used the “bool” type to represent boolean values which are either True or False. For instance, we use “bool” in the while loop when asking yes/no for whether the user completed their full vaccination to recognize invalid input. We have also used “int” in our program to represent those integers that are without fractional parts, which can be used to do calculations. And we used “float” to represent the floating number. The second data type we used is the list. We use the list to contain a series of numbers or strings. The list can be indexed and changed. You can change the specific element in the list and we store the ID number in the list. The third data type we used is the dictionary. The difference between dictionary and list is that dictionary has keys and values pairs, which can use key to find a value. For example, we can use the dictionary for administrators to change vaccination status. And we also use the string in our program. The string can be represented by single quotation ‘’ or double quotation “”. For example, we use a string when asking the user to input their id.

iv. Python implementation of the data types

The “bool” we have used in our program is designed to check if the ID number and password are correct when the administrator login. When the administrator inputs the

number and the password, the program will change the input into the boolean number to check whether it is correct. When asking for the date when the user got their vaccination, we used an integer to input the month and year. And we store the information input by the user in different lists. And the administrator can add information from the list by append() function. We use string to ask the user to input their information and form the retrieval information in proper order.

v. Modular design of the program:

1. Vaccination information input/storage/update/retrieval

We divide this part into six parts. First one inputs student/staffs' personal information, second part inputs their vaccination information, third part updates their personal information, fourth part updates their vaccination information, fifth part retrieves their personal information, and sixth part retrieves their vaccination information.

First one is *registration()* function, which lets users input their student/staff personal information, including student/staff ID, student/staff Name, their position in PolyU, department, vaccination status and password to login to the system. This *registration()* function calls other functions including *registration_id()*, *registration_name()*, *registration_position()*, *registration_department()*, *registration_vaccine_status()* and *password()* to receive input from users, also called *registration_confirm(id_number, name, position, department, vac_status)* function to confirm the information and give them opportunity to modify their information before write in *student_info.txt* file. We use commas to separate information when we write them in the text file, which is easier for later retrieval.

For the second part, we define a function called *write_vaccine_info(id_number, stuname, exist_id_list)*. This function allows users to input their type of vaccination, date of completing the full course of vaccination and the vaccination status of third dose covid-19 vaccine if applicable, and write them in the *vac_rec.txt* file. This function is called when users login successfully via the use of *student_staff_login()* function. This function accepts two variables, *id_number* and *stuname*, which is the id number and name of the person who login successfully. In this way, users no longer need to input their id numbers and names again, which could avoid some errors. Another variable this function accepts is *exist_id_list*, which is the id number that already existed in the vaccination file, to prevent people re-enter their information. This function calls *vaccine_name_input(id_number)* at first, and checks if the vaccine name inputted is CoronaVac. If it is, this function calls *input_year()*, *input_month()*, *input_day(year, month)* and *third_dose()* function to receive information from users. If not, this function also calls the above functions except *third_dose()* function, instead, it will write NA to this information. In *input_day(year, month)*, we use *calendar* module, which is also a python standard library, to generate a calendar according to the month and year chosen by the user for better view.

For the third part, we define *update_personal_info()* function. This function first reads the *student_info.txt* file and adds information in the text file to the dictionary using id numbers of them as keys. This function calls *update_personal_option_one(keys, info)* to update student/staffs' department information. First argument called in this function is the keys of the dictionary storing student/staff information. Second argument is the dictionary storing student/staff information. Second function *update_personal_option_two(keys, info)* allows the user to change student/staffs' vaccination statuses. Two arguments accepted in this function are exactly the same in the first function.

For the fourth part, we define *update_vaccine_info()* function. This function acts similarly as the *update_personal_info()* function. This function calls three functions, which are *update_vaccine_info_option_one(keys, ori_info)* to update type of vaccination, *update_vaccine_info_option_two(keys, ori_info)* to update the date of last dose of vaccination, *update_vaccine_info_option_three(keys, ori_info)* to update the vaccine status of third dose of vaccination. First argument accepted by these functions is the keys in the second argument, which is a dictionary storing the original information from the file *vac_rec.txt*. The difference between *update_personal_info()* and *update_vaccine_info()* is the first function updates things after the user finishes all changes he or she wants to change, however, the second one updates immediately when one change is made. We implement it in this way because everytime we need to read the updated version of information so that we could know if the third function could be executed.

For the fifth part, we define *personal_info_retrieval()* function. This function calls *personal_info_retrieval_option_one(keys, retrieval_info)* to retrieval information through student/staff id, *personal_info_retrieval_option_two(keys, retrieval_info)* to retrieval information through name, *personal_info_retrieval_option_three(keys, retrieval_info)* through department, *personal_info_retrieval_option_four(keys, retrieval_info)* to retrieve through position, and *personal_info_retrieval_option_five(keys, retrieval_info)* to retrieve through vaccination statuses. This function also calls another two functions to format the printing. First argument in these functions is the keys of a dictionary which stores information read from the file student_info.txt, and the second argument is the dictionary.

Sixth part is similar to the fifth part, the difference is we retrieve information from *vac_rec.txt* file. We define a function called *vaccine_retrieval()* and this function calls several other functions to retrieve information through student/staff ID, student/staff name, and type of vaccination. For the third function, we also print out the number of people in PolyU who receive this vaccination.

2. Student & staff login / Administrator login

As for student & staff login, we write a function *student_staff_login()* to enter a username and password. If the input is correct, the login is successful; if the ID number or password is empty, the ID number or password is empty, and it will display that ID number or password are empty; if the input exceeds three times, the input cannot be repeated.

We should read information from the file that contains students information and staff information at first, and check if the file is empty. Before reading the file, we assume the data of the ID number-*idn*, password-*psd*, vaccination status-*status*, student/staff name-*name* as lists. With information in the read file, we use the for loop to append information in lines of txt document into the list of the ID number, password, status, name (line 808-813). Then, the while true loop from line 814 makes this loop return to the beginning in case of an incorrect ID number or password, and the request continues.

In this login system, we need to check the number of login first because of the complexity of login system, so we count the number of login, assume count as 1, and count the number of login in line 816, if the input exceeds three times(*if count>4*), terminate this function. Then we check if the ID number and the password is in the file or not through the boolean value to see if the input is in the file. In this condition, we need to check the vaccination status to allow the user to input the vaccination status, if the vaccination status is yes, then let the user write the vaccine info and turn to the function *write_vaccine_info()*; if no, terminate this function. Then, check if the ID number and password is empty(convert to boolean value to see if the input is empty, and ask the user to input again.

Administrator login is quite similar to the student & staff login. We just have to turn the part *write_vaccine_info()* into the administrator option after login, return "Administrator Login Successfully" and check it in the function *main()*.

3. Vaccination Information Analysis

vaccine_analysis(): is responsible for this Vaccination information analysis can be divided into two parts, the first part is the percentage of the Vaccination Information. First of all, the admin must enter the information which needs to be analyzed first. Press 1 to know the percentage of total number(the number of polyu) to be completed or umcompleted.Press 2 to know the percentage of total number(the number of each department) to be completed or uncompleted. Press 3 to know the total number of polyu. Press 4 to know the total number of each department.Press 5 is the exit key.The id number of the person is used as the key.There are three buttons to choose from, press two of them for comparison or press button 4 to exit this interface. Button 1 can know the total number of polyu, button 2 can know the number of people who have completed the vaccine, and button 3 can know the number of people who have not completed the vaccine. Select two of the buttons for comparison and generate a percentage.The third step is to analyze the percentage of complete and uncompleted of a department.There are three options. Button 1 can know the total number of people in the department, button 2 can know the number of people who have completed vaccination in this department, and button 3 can know the number of people who have not finished vaccination in this department. Select two of the buttons to compare and form the percentage and the number of people.The last step is to analyze the percentage of complete and uncomplete polyu.There are three options. Button 1 can know the total number of people in polyu, button 2 can know the number of people who have completed vaccination in polyu, and button 3 can know the number of people who have not finished

vaccination in polyu. Select two of the buttons to compare and form the percentage and the number of people.

4. Email system(Extra function)

We propose this extra function because we noticed that some researchers suggest receiving a third dose of COVID-19 vaccination could provide a better immune response. HKSAR Government also announced the information of providing a third dose of vaccination recently. Our task here is to build a PolyU Vaccination Tracking System. We think it is appropriate if our system could also remind students of the updated vaccination information, simply like the department HSO in PolyU sends email with updated information of COVID-19 to us.

The selection of persons is achieved by using *compare()* function. This function extracts the information and compares the date, vaccination type and their status in third vaccination to find people fulfilling those conditions. In this function, we use the *datetime* module in Python. We convert the string representing the date of completing vaccination to a datetime object using the method *strptime()* and *timedelta()* in the datetime module to add 180 days to the date of completing vaccine. Still using this package, we read the date of today and compare both values. If today is later, the type of vaccination is CoronaVac, and status of third vaccination is No, *compare()* function will pass the email address to *email_sending(mail_receiver)* function and call it, which is responsible for sending email. Since here we don't know the real format of staff email, we assume the staff's email domain is the same as the student's, which is the id@connect.polyu.hk.

The *email_sending(mail_receiver)* function sends email automatically when receiving an email address passed by *compare()* function. This function uses two standard libraries in python, which are *smtplib* and *email*. *Smtplib* is responsible for sending email. In this program, we use the SMTP service provided by Gmail. And use MIMEText in *email.text.MIME* module to create an MIME object to construct the body of our email.

vi. Other special observation and approaches

One of the approaches we want to highlight is the importance of closing a file after use. During the implementation of function update vaccination information. We are very confused why the text file becomes empty when we execute the function and certainly write something to it. And when we terminate the program. Contents go back. Finally, we figure out we need to close the file every time after operation, so that the file will always update in time.