**Computer Science Project**

**Vaccine Proximity Database Management**

NAME :

CLASS:

ROLL

NUMBER:

**Index**

1. Problem Definition
2. Problem Analysis
3. Hardware and Software Requirement
4. Scope for Future Enhancement
5. Source Code
6. Output
7. Bibliography

**Problem Definition**

**Introduction**

‘Vaccination Planner’ is an app designed through the integration of Python, Python’s in-built module Tkinter and MySQL. This project was undertaken to increase the number of people vaccinated by making it easier for people to access the vaccine. We have utilized Python’s in-built Tkinter module for its user-friendly GUI with the record maintenance provisions available on MySQL.

**General Explanation**

* The app is divided into 2 major subdivisions – one for vaccine camps who administer the vaccine, the other for people who want to get their vaccine (the vaccinee).
* Before entering amidst the sub-categories, it is essential to note that this app in no way is entitled to any organization but is only a medium for the various vaccine camps (Hospitals, NGO’s etc.) to communicate with common people like us to ensure vaccination is provided at their disposal at the right time.
* When users open the app for the first time, they will be shown two options – Vaccinee and Vaccine Camp. Users are to choose according to their need.
* Choosing any of these options takes the user to a sign in and sign up page, where new users can create their account through the sign up page and existing users could log in to their account through the sign in page.
* While signing up users are shown the rules which they should follow to fill up their details.
* After users sign in to their accounts, they are shown their account details and the option to modify them if needed.
* Here, Vaccine Camp users can enter the vaccine stock available at their camp for the next day. They will not be allowed to enter or modify current day’s vaccine stock as all eligible vaccinee will already be notified about the vaccine availability.
* Here, Vaccinee users can check whether they are eligible to get the vaccine for that day, if they are eligible, they will be shown the details of the vaccine camp from where they could get their vaccine.
* The eligibility of a vaccinee is calculated by prioritizing who needs the vaccine most by considering factors such as the job they do, their age, past medical conditions, the vaccine they took for their first dose (if any) and the date they got their first dose.
* All the details entered by the users are stored in a MySQL database in a usable form. All other functions are done by python with the help of inputs from the MySQL server.

**Motive of the Project**

As we are all aware of the ongoing global epidemic COVID 19, vaccinations are the only way to keep oneself safe in this atmosphere of delusional virus. As informed citizens of the Indian country with lukewarm technological awareness we decided to make a software application which not only benefits and supervises the vaccination process but also guides both retailers and common people to establish a secure network of quick hassle-free vaccinations.

**Problem Analysis**

**Languages**

Programming languages used in the making of this project are as follows,

* Python
  + Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.
  + We made use of python to make use of its Tkinter GUI Module, take in input in a user-friendly way, run those input to get desired results and to display those result in a readable way to the user.
* MySQL
  + MySQL is an open-source relational database management system (RDBMS). A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data.
  + We made use of MySQL for storing all the vaccinee data, vaccine camp data, the priority list and also to manipulate these data hassle-free.

**Module**

The modules used in the program are,

* mysql.connector
  + As in this project we used MySQL to keep track of all the records and databases, we needed seamless connection between Python and MySQL. That was achieved with the help of mysl.connector .
  + This allowed us to enter new user data, vaccine data, filter records and update records.
* PIL
  + Python Imaging Library (PIL) is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats.
  + This was used to give a logo for our app.
* tkinter
  + Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI.
  + The clean and user-friendly user interface was made possible by the tkinter module.
* tkcalendar
  + This was used for getting dates from the users in an easy and intuitive way.

**Functions**

The functions defined and used in this app are listed below,

* connecting():

This function is used for establishing a connection between Python and MySQL server.

* rootwin():

This function is used to create the parent window for our app to work on and store some data which is required by the program.

* rootwin\_destroy():

This function is used to remove the widgets from the root window.

* dateformat (dfrmt):

This function is used to convert dates (‘dfrmt’) from yyyy-mm-dd to dd-mm-yyyy and return the value.

* sign\_frame(user='vaccinee', source='in', update='sign'):

This function is used to create all the required widgets for the sign in, sign up and update page for both vaccinee and vaccine camp users. It takes value ‘user’ to differentiate between vaccinee and vaccine camp users, ‘source' to differentiate whether to show sign in or sign up page and ‘update’ to know whether to show update page.

* sign\_frame\_destroy():

This function is used to remove widgets from the sign frame.

* signin\_button(user=’vaccinee’):

This function is used to check whether the entered sign in details are correct. If correct, redirects to the account information window. If not, shows a prompt to enter a valid set of details. It takes value ‘user’ to differentiate between vaccinee and vaccine camp users.

* signup\_button(user='vaccinee', update='sign'):

This function is used to create the widgets for the sign up or update page (depending upon the ‘update’ value), for user vaccinee or vaccine camps (as per the ‘user’ value).

* rbvacname\_cond():

This function is used to alter the signup or update page of the vaccinee by checking whether the vaccinee has taken their first dose.

* datacheck(user='vaccinee', update='sign'):

This function is used to check whether the entered details in the signup or update page (depending upon the ‘update’ value) are of the correct form, for user vaccinee or vaccine camps (as per the ‘user’ value).

* signupconfirm\_button(user='vaccinee', update='sign'):

This function is used to enter the details given by the vaccinee or vaccine camp (as per the ‘user’ value), from the sign up or update page (depending upon the ‘update’ value), into their respective tables in the MySQL database.

* priority(job,age,mh):

This function is used to determine the priority score for each vaccinee based on their user data. The priority list is made using this priority score. It is calculated by using the values ‘job’, ‘age’, ‘mh’. This list allows us to prioritize vaccine to the most vulnerable vaccinees at times of vaccine shortage.

* accinfo\_frame (user='vaccinee'):

This function is used to display all the details of the user vaccinee or vaccine camps (as per the ‘user’ value).

* vacsupply\_frame(user=’vaccinee’):

This function is used to show a vaccinee when they can get a vaccine, and allows a vaccine camp to enter how much vaccine they have available.

* updatevacsupply\_button():

This function is used to enter the details given by the vaccine camp regarding how much vaccine they have available into the MySQL database.

* mh(no):

This function is used to create a string of medical conditions from the stored form of medical conditions (‘no’) in the MySQL database and returns it.

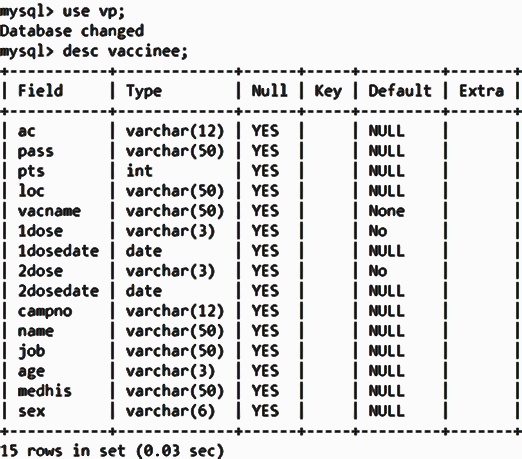
* vp():

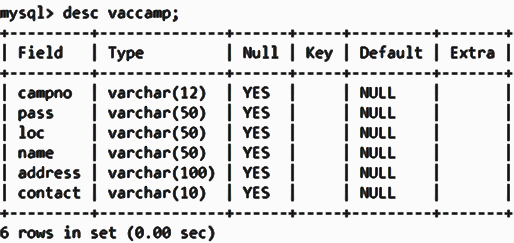
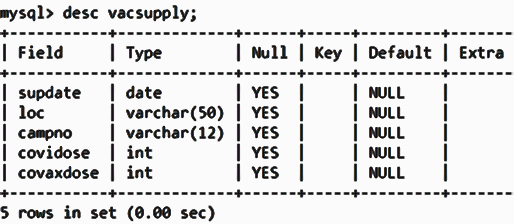
This function collects data from the MySQL database and determines which vaccinees get their vaccine for that day and stores that information in the database for later use.

**Working**

* First a root.mainloop() is established, to create a loop which gets input from the user interface.
* Then the connecting() function is called.
* Then the vp() function is called.
* Then the above mentioned user defined functions are called in the appropriate order to make the app work.

The following tables are created in a MySQL databse defined as ‘vp’,





**Hardware and Software Requirement**

**Minimum System Requirements**

* OS
  + Windows 7 or 10 or higher
  + Mac OS X 10.11 or higher
  + Linux RHEL 6/7 or higher
* Architecture: x86, 64-bit CPU (Intel / AMD Architecture)
* RAM: 4GB
* Free Disk Space: 8GB

**Scope for Future Enhancement**

**Modifications and Future Improvements**

* The prioritizing algorithm can be made even more sophisticated and accurate.
* If the supply of vaccine can be pre-determined better, we could schedule appointment months prior.
* The GUI could be updated to be even more sophisticated, and be visually pleasing. And be ported for many other applications involving appointment scheduling.
* Source code could be optimized in the future for better performance when a large number of users are using it at a time.
* Source could have been programmed to be even more modular and future proof. So that all modification needed in future could be implemented without changing the source code.
* Even more sophisticated and useful features can be implemented to make vaccination even more efficient.

**Source Code**

**#importing necessary modules**

**from tkinter import \* #for gui**

**from PIL import ImageTk, Image #for the program logo**

**from tkcalendar import DateEntry #for the interactive calendar**

**import mysql.connector as sql #for connecting mysql with python**

**#initialising root gui window**

**root=Tk()**

**#connecting to mysql server**

**def connecting():**

**global connection, cursor**

**connection=sql.connect(host='localhost',user='root',passwd='1234',database='vp')**

**cursor=connection.cursor(buffered=True)**

**rootwin()**

**#initialising gui**

**def rootwin():**

**global root, label\_welcome, button\_vaccinee, button\_vaccamp, curdate\_int, curdate\_str, curdate\_str\_ori, tmrwdate\_str, tmrwdate\_str\_ori**

**root.title('Vacination Planner')**

**root.iconbitmap('D:/RP/Study/Python/Python Codes/Project/Plus.ico')**

**root.resizable(0, 0)**

**label\_welcome=Label(root, text='Vaccine Proximity Database Management')**

**label\_welcome.grid(row=0, column=1)**

**button\_vaccinee=Button(root, text=' Vaccinee ', command=lambda:[sign\_frame(), sign\_frame(source='up')])**

**button\_vaccinee.grid(row=1, column=0, padx=10, pady=10)**

**button\_vaccamp=Button(root, text='Vaccine Camp', command=lambda:[sign\_frame(user='vaccamp'), sign\_frame(user='vaccamp',source='up')])**

**button\_vaccamp.grid(row=1, column=2, padx=10, pady=10)**

**#required values**

**exe='select current\_date'**

**cursor.execute(exe)**

**curdate\_str\_ori=str(cursor.fetchone()[0])**

**curdate\_int=int(curdate\_str\_ori[0:4]+curdate\_str\_ori[5:7]+curdate\_str\_ori[8:10])**

**curdate\_str=dateformat(curdate\_str\_ori)**

**exe='select date\_add("{}", interval 1 day)'.format(curdate\_str\_ori, )**

**cursor.execute(exe)**

**tmrwdate\_str\_ori=str(cursor.fetchone()[0])**

**tmrwdate\_str=dateformat(tmrwdate\_str\_ori)**

**#removing welcome page**

**def rootwin\_destroy():**

**label\_welcome.destroy()**

**button\_vaccinee.destroy()**

**button\_vaccamp.destroy()**

**#converts yyyy-mm-dd to dd-mm-yyyy**

**def dateformat(dfrmt):**

**return dfrmt[8:10]+'-'+dfrmt[5:7]+'-'+dfrmt[0:4]**

**#initialising vaccinee's and vaccine camp's signin and signup page**

**def sign\_frame(user='vaccinee', source='in', update='sign'):**

**global user\_signinid, user\_signinpass, frame\_signin, frame\_signup, button\_signup, button\_backroot**

**rootwin\_destroy()**

**#signin page**

**if source=='in':**

**#vaccinee signin page**

**frame\_signin\_txt='Vaccinee Sign In'**

**label\_signinid\_txt='Aadhar Number'**

**#vaccamp signin page**

**if user=='vaccamp':**

**frame\_signin\_txt='Vaccine Camp Sign In'**

**label\_signinid\_txt='Camp Number'**

**frame\_signin=LabelFrame(root, text=frame\_signin\_txt, padx=5, pady=5)**

**frame\_signin.grid(column=0, row=0, padx=10, pady=10)**

**user\_signinid=Entry(frame\_signin)**

**user\_signinid.grid(row=0, column=1, columnspan=2, padx=5, pady=5)**

**user\_signinpass=Entry(frame\_signin, show='\*')**

**user\_signinpass.grid(row=1, column=1, columnspan=2, padx=5, pady=5)**

**label\_signinid=Label(frame\_signin, text=label\_signinid\_txt)**

**label\_signinid.grid(row=0, column=0)**

**label\_signinpass=Label(frame\_signin, text='Password')**

**label\_signinpass.grid(row=1, column=0)**

**button\_signin=Button(frame\_signin, text='Sign In', command=lambda:signin\_button(user))**

**button\_signin.grid(row=2, column=0, padx=10, pady=10, columnspan=3)**

**#signup and update page**

**else:**

**#signup page**

**if update=='sign':**

**#vaccinee signup**

**frame\_signup\_txt='Vaccinee Sign Up'**

**#vaccamp signup**

**if user=='vaccamp':**

**frame\_signup\_txt='Vaccine Camp Sign Up'**

**frame\_signup=LabelFrame(root, text=frame\_signup\_txt, padx=5, pady=5)**

**frame\_signup.grid(column=0, row=1, padx=10, pady=10)**

**button\_signup=Button(frame\_signup, text=' Sign Up ', command=lambda:signup\_button(user))**

**button\_signup.grid(row=0, column=1, padx=10, pady=10)**

**button\_backroot=Button(frame\_signup, text=' Back ', command=lambda:[sign\_frame\_destroy(), rootwin()])**

**button\_backroot.grid(row=1, column=1, padx=10, pady=10)**

**#update page**

**else:**

**#vaccinee update page**

**frame\_signupdate\_txt='Vaccinee Update'**

**#vaccamp update page**

**if user=='vaccamp':**

**frame\_signupdate\_txt='Vaccine Camp Update'**

**frame\_signup=LabelFrame(root, text=frame\_signupdate\_txt, padx=5, pady=5)**

**frame\_signup.grid(column=0, row=1, padx=10,pady=10)**

**#signin and signup page**

**def sign\_frame\_destroy():**

**frame\_signin.destroy()**

**frame\_signup.destroy()**

**#code to be executed when the sigin button is pressed**

**def signin\_button(user='vaccinee'):**

**global cur\_user\_id**

**#step for vaccinee sigin process**

**if user=='vaccinee':**

**exe='select ac, pass from vaccinee'**

**#step for vaccamp sigin process**

**else:**

**exe='select campno, pass from vaccamp'**

**cursor.execute(exe)**

**creds=cursor.fetchall()**

**#checking entered details with details in databases**

**if (user\_signinid.get(), user\_signinpass.get()) in creds:**

**cur\_user\_id=user\_signinid.get()**

**sign\_frame\_destroy()**

**accinfo\_frame(user)**

**#message to be displayed when incorrect details entered**

**else:**

**incrt\_signin\_txt='Aadhar'**

**if user=='vaccamp':**

**incrt\_signin\_txt='Camp'**

**messagebox.showerror('Incorrect '+incrt\_signin\_txt+' Number or Password', 'Enter a valid 12 digit '+incrt\_signin\_txt+' Number or Check whether the entered details are correct.')**

**#code to be executed when the signup or update button is pressed**

**def signup\_button(user='vaccinee', update='sign'):**

**#vaccinee signup or update button**

**if user=='vaccinee':**

**global pat\_signupaadhar, pat\_signupname, pat\_signupage, pat\_signupmedhis1, pat\_signupmedhis2, pat\_signupmedhis3, pat\_signupmedhis4, pat\_signupmedhis5, pat\_signupmedhis6, pat\_signupmedhis7, pat\_signupmedhis8, pat\_signupmedhis9, pat\_signupmedhis10, pat\_signuppass, button\_signupconfirm, pat\_signupsex, pat\_signuploc, pat\_signupjob, pat\_signupvacdose, rbvacname\_cond1, rbvacname\_cond2**

**if update=='sign':**

**frame\_signin.destroy()**

**button\_signup.destroy()**

**button\_backroot.destroy()**

**#entries to get details from the user as text**

**pat\_signupaadhar=Entry(frame\_signup)**

**pat\_signupaadhar.grid(row=0,column=1,columnspan=2, padx=5, pady=5)**

**pat\_signupname=Entry(frame\_signup)**

**pat\_signupname.grid(row=1,column=1,columnspan=2, padx=5, pady=5)**

**#radiobutton to get details from the user as mcqs**

**list\_rbsex=[('Male','Male'),('Female','Female')]**

**pat\_signupsex=StringVar()**

**pat\_signupsex.set('Male')**

**rb\_sex\_count=0**

**for view\_sex, val\_sex in list\_rbsex:**

**Radiobutton(frame\_signup, text=view\_sex, variable=pat\_signupsex, value=val\_sex, command=lambda:None).grid(row=2+rb\_sex\_count,column=1,columnspan=2)**

**rb\_sex\_count+=1**

**pat\_signupage=Entry(frame\_signup)**

**pat\_signupage.grid(row=4,column=1,columnspan=2, padx=5, pady=5)**

**#drop down box to get details from the user from a list of options**

**list\_loc=['Tiruvallur','Sriperumbudur','Chennai North','Chennai South','Chennai Central']**

**pat\_signuploc=StringVar()**

**pat\_signuploc.set('Tiruvallur')**

**drop\_loc=OptionMenu(frame\_signup, pat\_signuploc, \*list\_loc)**

**drop\_loc.grid(row=5, column=1, columnspan=2, padx=5, pady=5)**

**list\_job=['Health Workers','Staffs of Congregate Settings','School Employee','Public Workers','IT','Others']**

**pat\_signupjob=StringVar()**

**pat\_signupjob.set('Health Workers')**

**drop\_job=OptionMenu(frame\_signup, pat\_signupjob, \*list\_job)**

**drop\_job.grid(row=6, column=1, columnspan=2, padx=5, pady=5)**

**#check button to get details from the user as check marks**

**pat\_signupmedhis1=IntVar()**

**pat\_signupmedhis2=IntVar()**

**pat\_signupmedhis3=IntVar()**

**pat\_signupmedhis4=IntVar()**

**pat\_signupmedhis5=IntVar()**

**pat\_signupmedhis6=IntVar()**

**pat\_signupmedhis7=IntVar()**

**pat\_signupmedhis8=IntVar()**

**pat\_signupmedhis9=IntVar()**

**pat\_signupmedhis10=IntVar()**

**Checkbutton(frame\_signup, text='Cancer', variable=pat\_signupmedhis1).grid(row=7, column=1)**

**Checkbutton(frame\_signup, text='Chronic Kidney Disease', variable=pat\_signupmedhis2).grid(row=8, column=1)**

**Checkbutton(frame\_signup, text='Chronic Lung Disease', variable=pat\_signupmedhis3).grid(row=9, column=1)**

**Checkbutton(frame\_signup, text='Neurological Conditions', variable=pat\_signupmedhis4).grid(row=10, column=1)**

**Checkbutton(frame\_signup, text='Diabetes', variable=pat\_signupmedhis5).grid(row=11, column=1)**

**Checkbutton(frame\_signup, text='Pregnancy', variable=pat\_signupmedhis6).grid(row=12, column=1)**

**Checkbutton(frame\_signup, text='Heart Conditions', variable=pat\_signupmedhis7).grid(row=13, column=1)**

**Checkbutton(frame\_signup, text='HIV Infection', variable=pat\_signupmedhis8).grid(row=14, column=1)**

**Checkbutton(frame\_signup, text='Weakend Immune System', variable=pat\_signupmedhis9).grid(row=15, column=1)**

**Checkbutton(frame\_signup, text='Liver Disease', variable=pat\_signupmedhis10).grid(row=16, column=1)**

**list\_medhis=['Cancer', 'Chronic Kidney Disease', 'Chronic Lung Disease', 'Neurological Conditions', 'Diabetes', 'Pregnancy', 'Heart Conditions', 'HIV Infection', 'Weakend Immune System', 'Liver Disease']**

**list\_rbvacdose=[('Yes','Yes'),('No','No')]**

**#to check whether to give the option of choosing 1st dose vaccine name and date**

**pat\_signupvacdose=StringVar()**

**pat\_signupvacdose.set('Yes')**

**rb\_vacdose\_count=0**

**for view\_vacdose, val\_vacdose in list\_rbvacdose:**

**Radiobutton(frame\_signup, text=view\_vacdose, variable=pat\_signupvacdose, value=val\_vacdose, command=lambda:None).grid(row=17+rb\_vacdose\_count,column=1,columnspan=2)**

**rb\_vacdose\_count+=1**

**rbvacname\_cond1=None**

**rbvacname\_cond2=True**

**rbvacname\_cond()**

**button\_checkvac=Button(frame\_signup, text='Check',command=rbvacname\_cond)**

**button\_checkvac.grid(row=18,column=3, padx=10, pady=10)**

**pat\_signuppass=Entry(frame\_signup, show='\*')**

**pat\_signuppass.grid(row=22,column=1,columnspan=2, padx=5, pady=5)**

**if update=='sign':**

**label\_ac=Label(frame\_signup, text='Aadhar Number')**

**label\_ac.grid(row=0,column=0)**

**label\_name=Label(frame\_signup, text='Name')**

**label\_name.grid(row=1,column=0)**

**label\_sex=Label(frame\_signup, text='Sex')**

**label\_sex.grid(row=2,column=0)**

**label\_age=Label(frame\_signup, text='Age')**

**label\_age.grid(row=4,column=0)**

**label\_loc=Label(frame\_signup, text='Location')**

**label\_loc.grid(row=5,column=0)**

**label\_job=Label(frame\_signup, text='Job')**

**label\_job.grid(row=6,column=0)**

**label\_medh=Label(frame\_signup, text='Medical History')**

**label\_medh.grid(row=7,column=0)**

**label\_vacdose=Label(frame\_signup, text='Whether 1st Dose Administered')**

**label\_vacdose.grid(row=17,column=0)**

**label\_pass=Label(frame\_signup, text='Password')**

**label\_pass.grid(row=22,column=0)**

**if update=='sign':**

**button\_signupconfirm=Button(frame\_signup, text='Confirm',command=lambda:datacheck())**

**button\_signupconfirm.grid(row=23,column=1,columnspan=2, padx=10, pady=10)**

**else:**

**button\_updateconfirm=Button(frame\_signup, text='Update',command=lambda:datacheck(update='update'))**

**button\_updateconfirm.grid(row=23,column=1,columnspan=2, padx=10, pady=10)**

**messagebox.showinfo('Note','-> Aadhar Number: Enter a valid 12 digit natural number.\n-> Name: Enter a valid name (should only contain alphabets and spaces, atleast one alphabet and not more than 50 characters)\n-> Age: Enter a valid integer from 18 to 125.\n-> Whether 1st Dose Administered: If No, select No and click Check. If Yes, select Yes and click Check.\n-> Date of 1st Dose: Enter a valid date from 16-01-2021 to '+curdate\_str+'.\n-> Password: Enter a valid password having a minimum of 8 characters and a maximum of 50 characters.')**

**if update=='sign':**

**button\_signin\_frsignup=Button(frame\_signup, text='Or Sign In',command=lambda:[sign\_frame\_destroy(), sign\_frame(), sign\_frame(source='up')])**

**button\_signin\_frsignup.grid(row=24,column=1, columnspan=2, padx=10, pady=10)**

**else:**

**button\_updatecancel=Button(frame\_signup, text='Cancel',command=lambda:[accinfo\_frame(), frame\_signup.destroy()])**

**button\_updatecancel.grid(row=24,column=1,columnspan=2, padx=10, pady=10)**

**#vaccamp signup or update button**

**else:**

**global vc\_signupcampno, vc\_signupname, vc\_signuploc, vc\_signupaddress, vc\_signupcontact, vc\_signuppass, vc\_button\_signupconfirm**

**if update=='sign':**

**frame\_signin.destroy()**

**button\_signup.destroy()**

**button\_backroot.destroy()**

**vc\_signupcampno=Entry(frame\_signup)**

**vc\_signupcampno.grid(row=0,column=1,columnspan=2, padx=5, pady=5)**

**vc\_signupname=Entry(frame\_signup)**

**vc\_signupname.grid(row=1,column=1,columnspan=2, padx=5, pady=5)**

**list\_loc=['Tiruvallur','Sriperumbudur','Chennai North','Chennai South','Chennai Central']**

**vc\_signuploc=StringVar()**

**vc\_signuploc.set('Tiruvallur')**

**drop\_loc=OptionMenu(frame\_signup, vc\_signuploc, \*list\_loc)**

**drop\_loc.grid(row=2, column=1, columnspan=2, padx=5, pady=5)**

**vc\_signupaddress=Entry(frame\_signup)**

**vc\_signupaddress.grid(row=3,column=1,columnspan=2, padx=5, pady=5)**

**vc\_signupcontact=Entry(frame\_signup)**

**vc\_signupcontact.grid(row=4, column=1, columnspan=2, padx=5, pady=5)**

**vc\_signuppass=Entry(frame\_signup, show='\*')**

**vc\_signuppass.grid(row=5,column=1,columnspan=2, padx=5, pady=5)**

**if update=='sign':**

**vc\_label\_campno=Label(frame\_signup, text='Camp Number')**

**vc\_label\_campno.grid(row=0,column=0)**

**vc\_label\_name=Label(frame\_signup, text='Name')**

**vc\_label\_name.grid(row=1,column=0)**

**vc\_label\_loc=Label(frame\_signup, text='Location')**

**vc\_label\_loc.grid(row=2,column=0)**

**vc\_label\_address=Label(frame\_signup, text='Address')**

**vc\_label\_address.grid(row=3,column=0)**

**vc\_label\_contact=Label(frame\_signup, text='Contact')**

**vc\_label\_contact.grid(row=4,column=0)**

**vc\_label\_pass=Label(frame\_signup, text='Password')**

**vc\_label\_pass.grid(row=5,column=0)**

**if update=='sign':**

**vc\_button\_signupconfirm=Button(frame\_signup, text='Confirm',command=lambda:datacheck(user='vaccamp'))**

**vc\_button\_signupconfirm.grid(row=6,column=1,columnspan=2, padx=10, pady=10)**

**else:**

**vc\_button\_updateconfirm=Button(frame\_signup, text='Update',command=lambda:datacheck(user='vaccamp', update='update'))**

**vc\_button\_updateconfirm.grid(row=6,column=1,columnspan=2, padx=10, pady=10)**

**messagebox.showinfo('Note','-> Camp Number: Enter a valid 12 digit natural number.\n-> Name: Enter a valid name (should only contain alphabets and spaces, atleast one alphabet and not more than 50 characters)\n-> Address: Enter a valid string atleast 1 character long and a maximum length of 100.\n-> Contact: Enter a valid 10 digit natural number.\n-> Password: Enter a valid password having a minimum of 8 characters and a maximum of 50 characters.')**

**if update=='sign':**

**vc\_button\_signin\_frsignup=Button(frame\_signup, text='Or Sign In',command=lambda:[sign\_frame\_destroy(), sign\_frame(user='vaccamp'), sign\_frame(user='vaccamp', source='up')])**

**vc\_button\_signin\_frsignup.grid(row=7,column=1, columnspan=2, padx=10, pady=10)**

**else:**

**vc\_button\_updatecancel=Button(frame\_signup, text='Cancel',command=lambda:[accinfo\_frame(user='vaccamp'), frame\_signup.destroy()])**

**vc\_button\_updatecancel.grid(row=7,column=1,columnspan=2, padx=10, pady=10)**

**#function to check whether to give the option of choosing 1st dose vaccine name and date**

**def rbvacname\_cond():**

**global rbvacname\_cond1, rbvacname\_cond2, label\_vacname, pat\_signupvacname, list\_rbvacname\_del, label\_vacdate, cal\_vacdate**

**if (pat\_signupvacdose.get()=='Yes' and rbvacname\_cond1!='Y') or rbvacname\_cond2:**

**label\_vacname=Label(frame\_signup, text='Vaccine Name')**

**label\_vacname.grid(row=19,column=0)**

**list\_rbvacname=[('Covishield','Covishield'),('Covaxin','Covaxin')]**

**pat\_signupvacname=StringVar()**

**pat\_signupvacname.set('Covishield')**

**rb\_vacname\_count=0**

**list\_rbvacname\_del=[]**

**for view\_vacname, val\_vacname in list\_rbvacname:**

**rb=Radiobutton(frame\_signup, text=view\_vacname, variable=pat\_signupvacname, value=val\_vacname, command=lambda:None)**

**rb.grid(row=19+rb\_vacname\_count,column=1,columnspan=2)**

**rb\_vacname\_count+=1**

**list\_rbvacname\_del.append(rb)**

**rbvacname\_cond1='Y'**

**rbvacname\_cond2=False**

**label\_vacdate=Label(frame\_signup, text='Date of 1st Dose')**

**label\_vacdate.grid(row=21,column=0, padx=5, pady=5)**

**cal\_vacdate=DateEntry(frame\_signup, locale='en\_US', date\_pattern='yyyy/MM/dd')**

**cal\_vacdate.grid(row=21,column=1,columnspan=2, padx=5, pady=5)**

**elif pat\_signupvacdose.get()=='No' and rbvacname\_cond1!='N' and rbvacname\_cond1!=None:**

**for w in list\_rbvacname\_del:**

**w.destroy()**

**label\_vacname.destroy()**

**label\_vacdate.destroy()**

**cal\_vacdate.destroy()**

**rbvacname\_cond1='N'**

**#to check whether the given data is in the correct form**

**def datacheck(user='vaccinee', update='sign'):**

**#checking data entered in vaccinee signup or update page**

**if user=='vaccinee':**

**exe='select ac from vaccinee;'**

**cursor.execute(exe)**

**db\_aadharlist=cursor.fetchall()**

**name\_cond=True**

**if pat\_signupname.get().strip()=='':**

**name\_cond=False**

**else:**

**for i in pat\_signupname.get().strip():**

**if i.isalpha() or (i==' '):**

**pass**

**else:**

**name\_cond=False**

**break**

**if pat\_signupvacdose.get()=='Yes':**

**giv\_date=str(cal\_vacdate.get\_date())**

**giv\_date=int(giv\_date[0:4]+giv\_date[5:7]+giv\_date[8:10])**

**if update=='sign' and (len(pat\_signupaadhar.get())!=12 or (not pat\_signupaadhar.get().isdigit())):**

**messagebox.showerror('Invalid Aadhar Number','Enter a valid 12 digit natural number')**

**elif update=='sign' and ((pat\_signupaadhar.get(),) in db\_aadharlist):**

**messagebox.showerror('Account already present', 'There is already an account created with the entered Aadhar Number')**

**elif len(pat\_signupname.get().strip())>50 or (not name\_cond):**

**messagebox.showerror('Invalid Name','Enter a valid name (should only contain alphabets and spaces, atleast one alphabet and not more than 50 characters)')**

**elif (not pat\_signupage.get().isdigit()) or int(pat\_signupage.get())>125 or int(pat\_signupage.get())<18 :**

**messagebox.showerror('Invalid Age','Enter a valid integer from 18 to 125')**

**elif pat\_signupvacdose.get()=='Yes' and (20210115>giv\_date or giv\_date>curdate\_int):**

**messagebox.showerror('Invalid Date', 'Enter a valid date from 16-01-2021 to '+curdate\_str+'.')**

**elif len(pat\_signuppass.get())<8 or len(pat\_signuppass.get())>50:**

**messagebox.showerror('Invalid Password','Enter a valid password having a minimum of 8 characters and a maximum of 50 characters')**

**else:**

**signupconfirm\_button(user, update)**

**#checking data entered in vaccamp signup or update page**

**else:**

**exe='select campno from vaccamp'**

**cursor.execute(exe)**

**db\_campnolist=cursor.fetchall()**

**vc\_name\_cond=True**

**if vc\_signupname.get().strip()=='':**

**vc\_name\_cond=False**

**else:**

**for i in vc\_signupname.get().strip():**

**if i.isalpha() or (i==' '):**

**pass**

**else:**

**vc\_name\_cond=False**

**break**

**if update=='sign' and (len(vc\_signupcampno.get())!=12 or (not vc\_signupcampno.get().isdigit())):**

**messagebox.showerror('Invalid Camp Number','Enter a valid 12 digit natural number')**

**elif update=='sign' and ((vc\_signupcampno.get(),) in db\_campnolist):**

**messagebox.showerror('Account already present', 'There is already an account created with the entered Camp Number')**

**elif len(vc\_signupname.get().strip())>50 or (not vc\_name\_cond):**

**messagebox.showerror('Invalid Name','Enter a valid name (should only contain alphabets and spaces, atleast one alphabet and not more than 50 characters)')**

**elif len(vc\_signupaddress.get().strip())>100 or len(vc\_signupaddress.get().strip())==0:**

**messagebox.showerror('Invalid Address','Enter a valid string atleast 1 character long and a maximum length of 100')**

**elif (len(vc\_signupcontact.get())!=10 or (not vc\_signupcontact.get().isdigit())):**

**messagebox.showerror('Invalid Contact','Enter a valid 10 digit natural number')**

**elif len(vc\_signuppass.get())<8 or len(vc\_signuppass.get())>50:**

**messagebox.showerror('Invalid Password','Enter a valid password having a minimum of 8 characters and a maximum of 50 characters')**

**else:**

**signupconfirm\_button(user, update)**

**#updating entered data in signup or update page into mysql database**

**def signupconfirm\_button(user='vaccinee', update='sign'):**

**#entering data in vaccinee signup or update page into mysql database**

**if user=='vaccinee':**

**#medical history stored as ones and zeroes for simplicity**

**pat\_signupmedhis=str(pat\_signupmedhis1.get())+str(pat\_signupmedhis2.get())+str(pat\_signupmedhis3.get())+str(pat\_signupmedhis4.get())+str(pat\_signupmedhis5.get())+str(pat\_signupmedhis6.get())+str(pat\_signupmedhis7.get())+str(pat\_signupmedhis8.get())+str(pat\_signupmedhis9.get())+str(pat\_signupmedhis10.get())**

**job\_dict={'Health Workers':1,'Staffs of Congregate Settings':2,'School Employee':3,'Public Workers':4,'IT':5,'Others':6}**

**priority(job\_dict[pat\_signupjob.get()],int(pat\_signupage.get()),pat\_signupmedhis.count('1'))**

**#updating entered data in vaccinee update page into mysql database**

**if update=='update':**

**#updating entered data in vaccinee update page into mysql database where the vaccinee did get their 1st dose**

**if pat\_signupvacdose.get()=='Yes':**

**cal\_vacdate1=str(cal\_vacdate.get\_date())**

**exe='select date\_add("{}", interval 84 day)'.format(str(cal\_vacdate.get\_date()),)**

**cursor.execute(exe)**

**cal\_vacdate2=cursor.fetchone()[0]**

**exe='update vaccinee set pass="{}", pts={}, loc="{}", vacname="{}", 1dose="{}", 1dosedate="{}", 2dosedate="{}", name="{}", job="{}", age="{}", medhis="{}", sex="{}" where ac="{}";'.format(pat\_signuppass.get(), pts, pat\_signuploc.get(), pat\_signupvacname.get(), pat\_signupvacdose.get(), cal\_vacdate1, cal\_vacdate2, pat\_signupname.get().strip(), pat\_signupjob.get(), pat\_signupage.get(), pat\_signupmedhis, pat\_signupsex.get(), cur\_user\_id)**

**cursor.execute(exe)**

**#updating entered data in vaccinee update page into mysql database where the vaccinee didn't get their 1st dose**

**elif pat\_signupvacdose.get()=='No':**

**exe='update vaccinee set pass="{}", pts={}, loc="{}", name="{}", job="{}", age="{}", medhis="{}", sex="{}" where ac="{}";'.format(pat\_signuppass.get(), pts, pat\_signuploc.get(), pat\_signupname.get().strip(), pat\_signupjob.get(), pat\_signupage.get(), pat\_signupmedhis, pat\_signupsex.get(), cur\_user\_id)**

**cursor.execute(exe)**

**#entering data in vaccinee update page into mysql database**

**else:**

**#entering data in vaccinee update page into mysql database where the vaccinee did get their 1st dose**

**if pat\_signupvacdose.get()=='Yes':**

**cal\_vacdate1=str(cal\_vacdate.get\_date())**

**exe='select date\_add("{}", interval 84 day)'.format(str(cal\_vacdate.get\_date()),)**

**cursor.execute(exe)**

**cal\_vacdate2=cursor.fetchone()[0]**

**attributes=(pat\_signupaadhar.get() ,pat\_signuppass.get(), pts, pat\_signuploc.get(), pat\_signupvacname.get(), pat\_signupvacdose.get(), cal\_vacdate1, cal\_vacdate2, pat\_signupname.get().strip(), pat\_signupjob.get(), pat\_signupage.get(), pat\_signupmedhis, pat\_signupsex.get())**

**exe='insert into vaccinee (ac, pass, pts, loc, vacname, 1dose, 1dosedate, 2dosedate, name, job, age, medhis, sex) values {};'.format(attributes,)**

**cursor.execute(exe)**

**#entering data in vaccinee update page into mysql database where the vaccinee didn't get their 1st dose**

**elif pat\_signupvacdose.get()=='No':**

**attributes=(pat\_signupaadhar.get() , pat\_signuppass.get(), pts, pat\_signuploc.get(), pat\_signupname.get().strip(), pat\_signupjob.get(), pat\_signupage.get(), pat\_signupmedhis, pat\_signupsex.get())**

**exe='insert into vaccinee (ac, pass, pts, loc, name, job, age, medhis, sex) values {};'.format(attributes,)**

**cursor.execute(exe)**

**#going back to previous page**

**if update=='update':**

**messagebox.showinfo('Info', 'Updated Details added to Database')**

**frame\_signup.destroy()**

**accinfo\_frame()**

**else:**

**messagebox.showinfo('Info', 'Details added to Database')**

**frame\_signup.destroy()**

**sign\_frame()**

**sign\_frame(source='up')**

**#running vaccine distribution program again for new user**

**vp()**

**#entering data in vaccamp signup or update page into mysql database**

**else:**

**#updating entered data in vaccamp update page into mysql database**

**if update=='update':**

**exe='update vaccamp set pass="{}", loc="{}", name="{}", address="{}", contact="{}" where campno="{}";'.format(vc\_signuppass.get(), vc\_signuploc.get(), vc\_signupname.get().strip(), vc\_signupaddress.get(), vc\_signupcontact.get(), cur\_user\_id)**

**cursor.execute(exe)**

**#entering data in vaccamp signup page into mysql database**

**else:**

**attributes=(vc\_signupcampno.get(), vc\_signuppass.get(), vc\_signuploc.get(), vc\_signupname.get().strip(), vc\_signupaddress.get(), vc\_signupcontact.get())**

**exe='insert into vaccamp (campno, pass, loc, name, address, contact) values {};'.format(attributes,)**

**cursor.execute(exe)**

**#going back to previous page**

**if update=='update':**

**messagebox.showinfo('Info', 'Updated Details added to Database')**

**frame\_signup.destroy()**

**accinfo\_frame(user='vaccamp')**

**else:**

**messagebox.showinfo('Info', 'Details added to Database')**

**frame\_signup.destroy()**

**sign\_frame(user='vaccamp')**

**sign\_frame(user='vaccamp', source='up')**

**#ordering table vaccinee in mysql as per their priority**

**exe='alter table vaccinee order by pts DESC'**

**cursor.execute(exe)**

**connection.commit()**

**#gives scores according to priority**

**def priority(job,age,mh):**

**global pts**

**pts=0**

**crit=0**

**if job==1:**

**pts+=12000**

**crit=1**

**elif age>=75:**

**pts+=11000+age-74**

**crit=2**

**elif 65<=age<=74 and mh>=2:**

**pts+=10000+age-64+((mh-1)\*10)**

**crit=2**

**elif job==2:**

**pts+=9000**

**crit=1**

**elif age>=65 and mh>=1:**

**pts+=8000+age-64+(mh\*10)**

**crit=2**

**elif age>=65:**

**pts+=7000+age-64**

**crit=2**

**elif mh>=2:**

**pts+=6000+((mh-1)\*10)**

**crit=2**

**elif job==3:**

**pts+=5000**

**crit=1**

**elif job==4:**

**pts+=4000**

**crit=1**

**elif 18<=age<=64 and mh>=1:**

**pts+=3000+age-15+(mh\*10)**

**crit=2**

**elif job==5:**

**pts+=2000**

**crit=1**

**elif job==6:**

**pts+=1000**

**crit=1**

**if crit==2:**

**if job==1:**

**pts+=600**

**elif job==2:**

**pts+=500**

**elif job==3:**

**pts+=400**

**elif job==4:**

**pts+=300**

**elif job==5:**

**pts+=200**

**elif job==6:**

**pts+=100**

**elif crit==1:**

**if age>=75:**

**pts+=600+age-74**

**elif 65<=age<=74 and mh>=2:**

**pts+=500+age-64+((mh-1)\*10)**

**elif age>=65 and mh>=1:**

**pts+=400+age-64+(mh\*10)**

**elif age>=65:**

**pts+=300+age-64**

**elif mh>=2:**

**pts+=200+((mh-1)\*10)**

**elif 18<=age<=64 and mh>=1:**

**pts+=100+age-15+(mh\*10)**

**#to display account info of users**

**def accinfo\_frame(user='vaccinee'):**

**global frame\_accinfo**

**#frame name for vaccinee**

**if user=='vaccinee':**

**frame\_accinfo\_txt='Vaccinee Acount Information'**

**exe='select \* from vaccinee where ac="{}";'.format(cur\_user\_id, )**

**#frame name for vaccamp**

**else:**

**frame\_accinfo\_txt='Vaccine Camp Acount Information'**

**exe='select \* from vaccamp where campno="{}";'.format(cur\_user\_id, )**

**cursor.execute(exe)**

**cur\_user\_info=cursor.fetchone()**

**frame\_accinfo=LabelFrame(root, text=frame\_accinfo\_txt, padx=5, pady=5)**

**frame\_accinfo.grid(padx=10,pady=10)**

**#vaccinee details**

**if user=='vaccinee':**

**#to convert ones and zeroes of medical history back into strings**

**mh\_str=mh(cur\_user\_info[13])**

**if mh\_str=='':**

**mh\_str='None'**

**label\_accvaccinee1=Label(frame\_accinfo, text='\nAadhar Number: '+cur\_user\_info[0]+'\n\nName: '+cur\_user\_info[10]+'\n\nSex: '+cur\_user\_info[14]+'\n\nAge: '+cur\_user\_info[12]+'\n\nLocation: '+cur\_user\_info[3]+'\n\nJob: '+cur\_user\_info[11]+'\n\nMedical History: '+mh\_str)**

**label\_accvaccinee1.grid()**

**if cur\_user\_info[7]=='Yes':**

**label\_accvaccinee2=Label(frame\_accinfo, text='\nVaccine Status: '+cur\_user\_info[4]+' 2nd Dose Completed'+'\n\nDate of 1st Dose: '+dateformat(str(cur\_user\_info[6]))+'\n\nDate of 2nd Dose: '+dateformat(str(cur\_user\_info[8])))**

**elif cur\_user\_info[5]=='Yes':**

**label\_accvaccinee2=Label(frame\_accinfo, text='\nVaccine Status: '+cur\_user\_info[4]+' 1st Dose Completed'+'\n\nDate of 1st Dose: '+dateformat(str(cur\_user\_info[6]))+'\n\nEligibile for 2nd Dose from: '+dateformat(str(cur\_user\_info[8]))+'\n')**

**else:**

**label\_accvaccinee2=Label(frame\_accinfo, text='\nVaccine Status: Needs to get 1st Dose\n')**

**label\_accvaccinee2.grid()**

**#user should not update after getting vaccine even once with app**

**exe='select campno from vaccinee where ac="{}";'.format(cur\_user\_id,)**

**cursor.execute(exe)**

**exist\_vaccinee=cursor.fetchone()[0]**

**if exist\_vaccinee==None:**

**button\_accinfoupdate=Button(frame\_accinfo, text='Update Details', command=lambda:[frame\_accinfo.destroy(),sign\_frame(source='up', update='update'),signup\_button(update='update')])**

**button\_accinfoupdate.grid(padx=10, pady=10)**

**button\_accinfocheck=Button(frame\_accinfo, text='Check for Vaccine', command=lambda:[vacsupply\_frame()])**

**button\_accinfocheck.grid(padx=10, pady=10)**

**button\_accinfologout=Button(frame\_accinfo, text='Log Out',command=lambda:[messagebox.showinfo('Info','Successfully logged out'), sign\_frame(), sign\_frame(source='up'), frame\_accinfo.destroy()])**

**button\_accinfologout.grid(padx=10, pady=10)**

**#vaccamp details**

**else:**

**label\_accvaccamp=Label(frame\_accinfo, text='\nCamp Number: '+cur\_user\_info[0]+'\n\nName: '+cur\_user\_info[3]+'\n\nLocation: '+cur\_user\_info[2]+'\n\nAddress: '+cur\_user\_info[4]+'\n\nContact: '+cur\_user\_info[5]+'\n')**

**label\_accvaccamp.grid()**

**button\_accinfoupdate=Button(frame\_accinfo, text='Update Details',command=lambda:[frame\_accinfo.destroy(),sign\_frame(user='vaccamp', source='up', update='update'),signup\_button(user='vaccamp', update='update')])**

**button\_accinfoupdate.grid(padx=10, pady=10)**

**vc\_button\_accinfosupply=Button(frame\_accinfo, text='Update Vaccine Stock', command=lambda:[vacsupply\_frame(user='vaccamp')])**

**vc\_button\_accinfosupply.grid(padx=10, pady=10)**

**button\_accinfologout=Button(frame\_accinfo, text='Log Out',command=lambda:[messagebox.showinfo('Info','Successfully logged out'), sign\_frame(user='vaccamp'), sign\_frame(user='vaccamp', source='up'), frame\_accinfo.destroy()])**

**button\_accinfologout.grid(padx=10, pady=10)**

**#frame for checking or entering vaccinee availability**

**def vacsupply\_frame(user='vaccinee'):**

**#frame for vaccinee to check vaccine availability**

**if user=='vaccinee':**

**frame\_accinfo.destroy()**

**frame\_vacsupply=LabelFrame(root, text='Vaccine Availability', padx=5, pady=5)**

**frame\_vacsupply.grid(padx=10, pady=10)**

**exe='select 1dose, 1dosedate, 2dose, 2dosedate, campno, vacname from vaccinee where ac="{}";'.format(cur\_user\_id, )**

**cursor.execute(exe)**

**vac\_status=cursor.fetchone()**

**exe='select loc, name, address, contact from vaccamp where campno="{}";'.format(vac\_status[4], )**

**cursor.execute(exe)**

**vc\_info=cursor.fetchone()**

**if vac\_status[0]=='No':**

**txt='\n1st Dose Vaccine Not Available Yet. Try again tomorrow\n'**

**elif vac\_status[0]=='Yes' and vac\_status[2]=='No':**

**if vac\_status[4]==None:**

**txt='\n2nd Dose Vaccine Not Available Yet. Try again tomorrow\n'**

**else:**

**txt='\n1st Dose of '+vac\_status[5]+' at Camp '+vac\_status[4]+' on '+str(vac\_status[1])+'\n\nCamp Info\n\nName: '+vc\_info[1]+'\n\nLocation: '+vc\_info[0]+'\n\nAddress: '+vc\_info[2]+'\n\nContact: '+vc\_info[3]+'\n'**

**elif vac\_status[0]=='Yes' and vac\_status[2]=='Yes':**

**txt='\n2nd Dose of '+vac\_status[5]+' at Camp '+vac\_status[4]+' on '+str(vac\_status[3])+'\n\nCamp Info\n\nName: '+vc\_info[1]+'\n\nLocation: '+vc\_info[0]+'\n\nAddress: '+vc\_info[2]+'\n\nContact: '+vc\_info[3]+'\n'**

**label\_vacstat=Label(frame\_vacsupply, text=txt)**

**label\_vacstat.grid()**

**button\_back=Button(frame\_vacsupply, text='Back', command=lambda:[frame\_vacsupply.destroy(), accinfo\_frame()])**

**button\_back.grid(padx=10, pady=10)**

**#frame for vaccamp to enter vaccine stock**

**else:**

**global vc\_covidose, vc\_covaxdose, vc\_frame\_vacsupply**

**frame\_accinfo.destroy()**

**vc\_frame\_vacsupply=LabelFrame(root, text='Vaccine Stock', padx=5, pady=5)**

**vc\_frame\_vacsupply.grid(padx=10, pady=10)**

**exe='select supdate, covidose, covaxdose from vacsupply where (campno="{}" and (supdate="{}" or supdate="{}"));'.format(cur\_user\_id, curdate\_str\_ori, tmrwdate\_str\_ori)**

**cursor.execute(exe)**

**existing\_sup=cursor.fetchall()**

**tocovi=tocovax=tmrwcovi=tmrwcovax=0**

**for rec in existing\_sup:**

**if str(rec[0])==curdate\_str\_ori:**

**tocovi=rec[1]**

**tocovax=rec[2]**

**elif str(rec[0])==tmrwdate\_str\_ori:**

**tmrwcovi=rec[1]**

**tmrwcovax=rec[2]**

**vc\_label\_exist=Label(vc\_frame\_vacsupply, text='\nVaccine Supply for the day '+curdate\_str+':\n\nCovishield: '+str(tocovi)+'\n\nCovaxin: '+str(tocovax)+'\n\n\nVaccine Supply for the day '+tmrwdate\_str+':\n\nCovishield: '+str(tmrwcovi)+'\n\nCovaxin: '+str(tmrwcovax)+'\n\n')**

**vc\_label\_exist.grid(row=0, column=0, columnspan=3)**

**vc\_label\_vacsupply=Label(vc\_frame\_vacsupply, text='Enter vaccine supply details for the day '+tmrwdate\_str+':')**

**vc\_label\_vacsupply.grid(row=1, column=0, columnspan=3)**

**vc\_covidose=Entry(vc\_frame\_vacsupply)**

**vc\_covidose.grid(row=2,column=1,columnspan=2, padx=5, pady=5)**

**vc\_covaxdose=Entry(vc\_frame\_vacsupply)**

**vc\_covaxdose.grid(row=3,column=1,columnspan=2, padx=5, pady=5)**

**vc\_label\_covidose=Label(vc\_frame\_vacsupply, text='Covishield')**

**vc\_label\_covidose.grid(row=2, column=0)**

**vc\_label\_covaxdose=Label(vc\_frame\_vacsupply, text='Covaxin')**

**vc\_label\_covaxdose.grid(row=3, column=0)**

**vc\_button\_updatevacsupply=Button(vc\_frame\_vacsupply, text='Update Stock', command=updatevacsupply\_button)**

**vc\_button\_updatevacsupply.grid(columnspan=3, padx=10, pady=10)**

**messagebox.showwarning('Important',"Could only update supply details for tomorrow. Changes of supply for tomorrow should be made today itself. Tomorrow's supply can't be changed tomorrow.")**

**vc\_button\_cancelvacsupply=Button(vc\_frame\_vacsupply, text='Cancel', command=lambda:[vc\_frame\_vacsupply.destroy(), accinfo\_frame(user='vaccamp')])**

**vc\_button\_cancelvacsupply.grid(columnspan=3, padx=10, pady=10)**

**#entering vaccine stocks to table vacsupply**

**def updatevacsupply\_button():**

**#checking whether entered details are correct**

**if (not vc\_covidose.get().isdigit()):**

**messagebox.showerror('Invalid Covishield Dose','Enter a valid integer')**

**elif (not vc\_covaxdose.get().isdigit()):**

**messagebox.showerror('Invalid Covaxin Dose','Enter a valid integer')**

**else:**

**exe='select campno from vacsupply where supdate="{}";'.format(tmrwdate\_str\_ori,)**

**cursor.execute(exe)**

**existing\_vc=cursor.fetchall()**

**#updating vaccine supply for an existing supply date**

**if (cur\_user\_id, ) in existing\_vc:**

**messagebox.showinfo('Info','Details updated to database successfully')**

**exe='update vacsupply set covidose={}, covaxdose={} where (supdate="{}" and campno="{}");'.format(int(vc\_covidose.get()), int(vc\_covaxdose.get()), tmrwdate\_str\_ori, cur\_user\_id)**

**cursor.execute(exe)**

**connection.commit()**

**#entering fresh vaccine supply for a supply date**

**else:**

**messagebox.showinfo('Info','Details added to database successfully')**

**exe='select loc from vaccamp where campno="{}";'.format(cur\_user\_id, )**

**cursor.execute(exe)**

**vc\_loc=cursor.fetchone()[0]**

**vac\_dose=(tmrwdate\_str\_ori, vc\_loc, cur\_user\_id, int(vc\_covidose.get()), int(vc\_covaxdose.get()))**

**exe='insert into vacsupply values {};'.format(vac\_dose, )**

**cursor.execute(exe)**

**connection.commit()**

**vc\_frame\_vacsupply.destroy()**

**accinfo\_frame(user='vaccamp')**

**#running vaccine distribution program again for updated vaccine stock**

**vp()**

**#converts medicl history ones and zeroes to strings**

**def mh(no):**

**list\_medhis=['Cancer', 'Chronic Kidney Disease', 'Chronic Lung Disease', 'Neurological Conditions', 'Diabetes', 'Pregnancy', 'Heart Conditions', 'HIV Infection', 'Weakend Immune System', 'Liver Disease']**

**s=''**

**for i in range(len(no)):**

**if no[i]=='1':**

**s+=list\_medhis[i]+', '**

**return s[:len(s)-2]**

**#function to distribute vaccines to vaccinee according to priority and vaccine availability**

**def vp():**

**exe='select loc, campno, covidose, covaxdose from vacsupply where supdate="{}";'.format(curdate\_str\_ori,)**

**cursor.execute(exe)**

**list\_vacsupply=cursor.fetchall()**

**exe='select loc, ac, vacname from vaccinee where (1dose="No" or (2dose="No" and 2dosedate<=current\_date()));'**

**cursor.execute(exe)**

**list\_vaccinee=cursor.fetchall()**

**#to stop function if no eligible vaccinees are present**

**if list\_vaccinee==[]:**

**return**

**#creating dictionary where key is location and value is details of vaccamp in given location**

**dict\_vacsupply={}**

**for rec in list\_vacsupply:**

**if rec[0] not in dict\_vacsupply:**

**dict\_vacsupply[rec[0]]=[list(rec[1:4])]**

**else:**

**dict\_vacsupply[rec[0]].append(list(rec[1:4]))**

**#creating dictionary where key is location and value is details of vaccinee in given location**

**dict\_vaccinee={}**

**for rec in list\_vaccinee:**

**val\_vaccinee=list(rec[1:3])**

**val\_vaccinee.append('No')**

**if rec[0] not in dict\_vaccinee:**

**dict\_vaccinee[rec[0]]=[val\_vaccinee]**

**else:**

**dict\_vaccinee[rec[0]].append(val\_vaccinee)**

**#loop for vaccine distribution**

**for loc in dict\_vacsupply:**

**for vc in dict\_vacsupply[loc]:**

**if loc in dict\_vaccinee:**

**for rec in dict\_vaccinee[loc]:**

**if rec[2]=='Yes':**

**continue**

**novac=False**

**if rec[1]=='Covishield':**

**index=1**

**elif rec[1]=='Covaxin':**

**index=2**

**else:**

**novac=True**

**if not novac and vc[index]-1>=0:**

**exe='update vaccinee set 2dose="Yes", 2dosedate="{}", campno="{}" where ac="{}";'.format(curdate\_str\_ori, vc[0], rec[0])**

**cursor.execute(exe)**

**connection.commit()**

**vc[index]=vc[index]-1**

**rec[2]=('Yes')**

**elif novac:**

**for index in [1, 2]:**

**if vc[index]-1>=0:**

**if index==1:**

**vac='Covishield'**

**else:**

**vac='Covaxin'**

**exe='select date\_add("{}", interval 84 day)'.format(curdate\_str\_ori,)**

**cursor.execute(exe)**

**vacdate2=str(cursor.fetchone()[0])**

**exe='update vaccinee set vacname="{}", 1dose="Yes", 1dosedate="{}", 2dosedate="{}", campno="{}" where ac="{}";'.format(vac, curdate\_str\_ori, vacdate2, vc[0], rec[0])**

**cursor.execute(exe)**

**connection.commit()**

**vc[index]=vc[index]-1**

**rec[2]=('Yes')**

**break**

**#updating vacsupply for accurate results when function is run again**

**exe='update vacsupply set covidose={}, covaxdose={} where (campno="{}" and supdate="{}");'.format(vc[1], vc[2], vc[0], curdate\_str\_ori)**

**cursor.execute(exe)**

**connection.commit()**

**connecting()**

**#running vaccine distribution program**

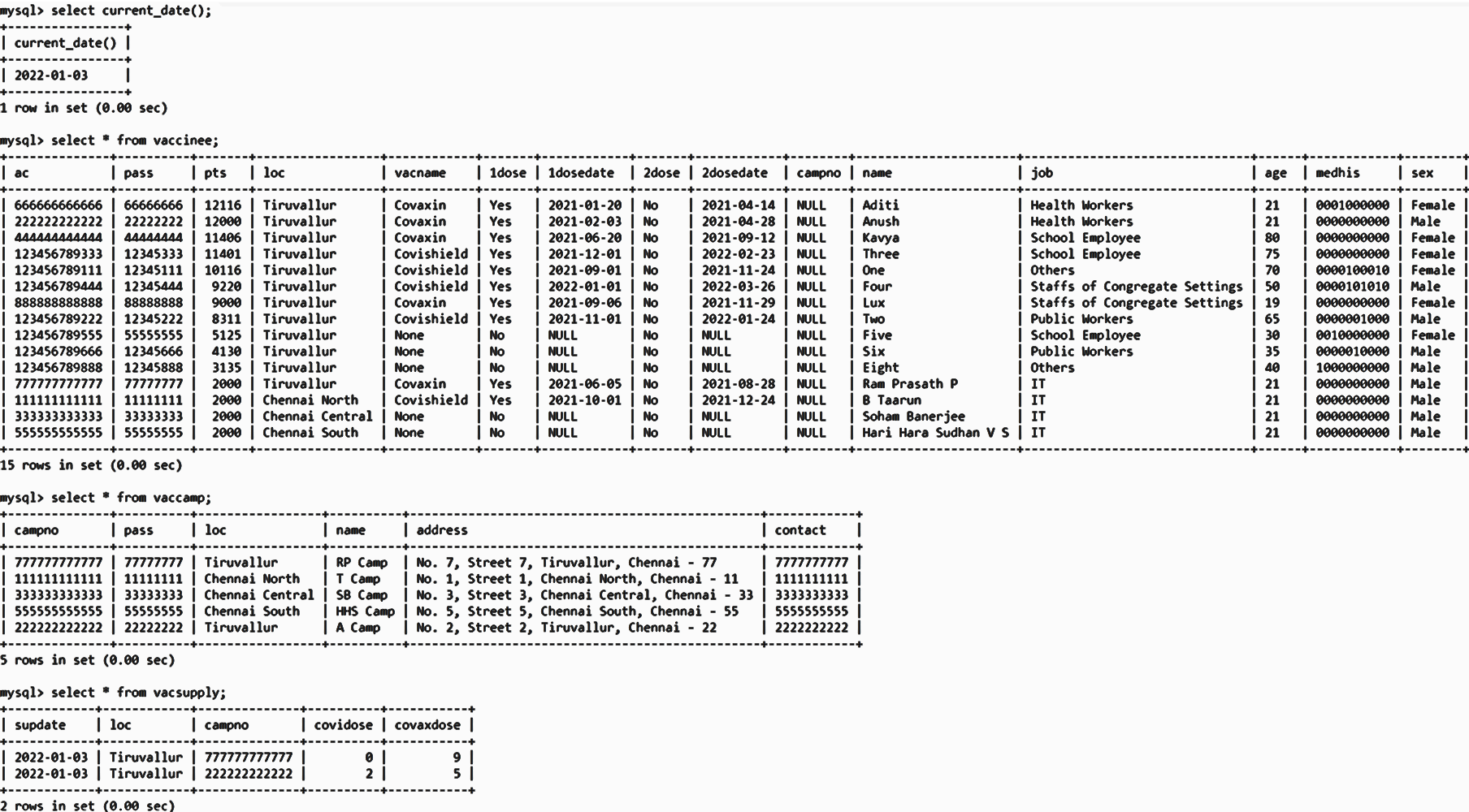
**vp()**

**#starting tkinter loop to interact with the gui**

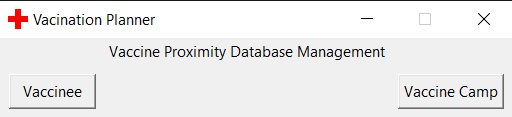
**root.mainloop()**

**Output**

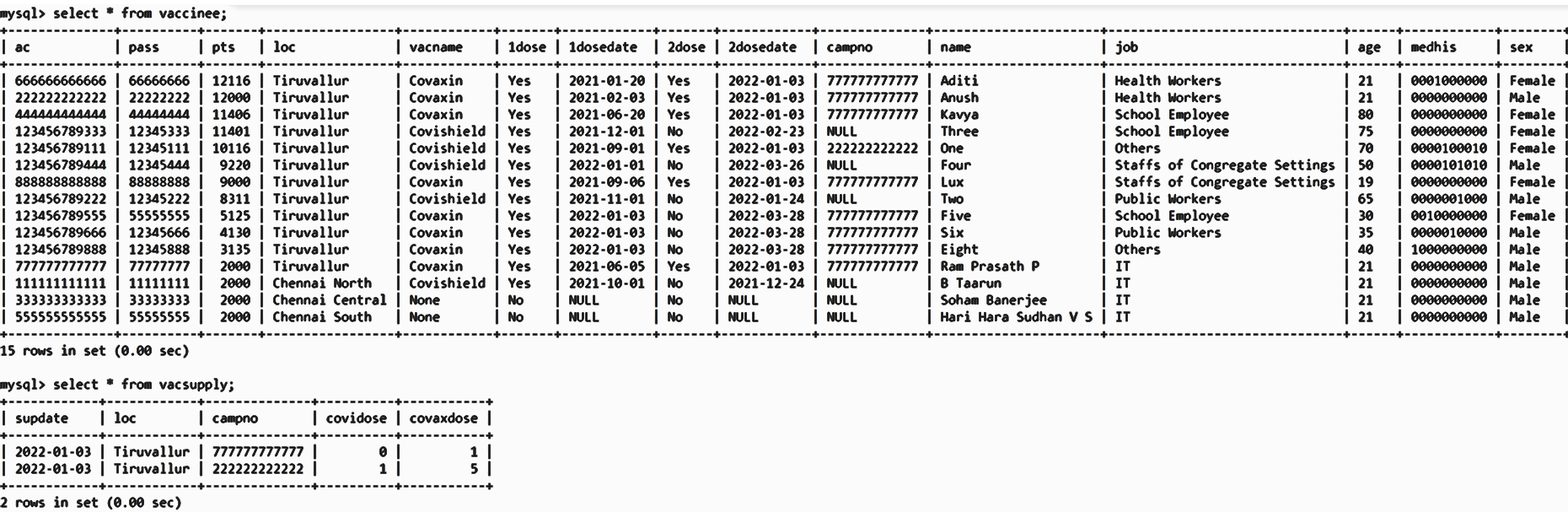
* The MySQL database ‘vp’ has the following tables,



* When the program is run, the connecting() function is called,

****

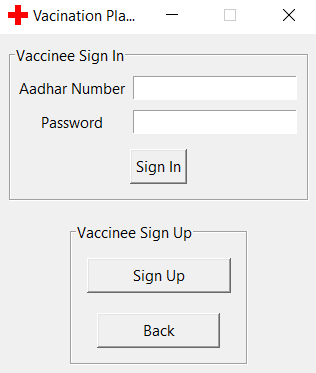
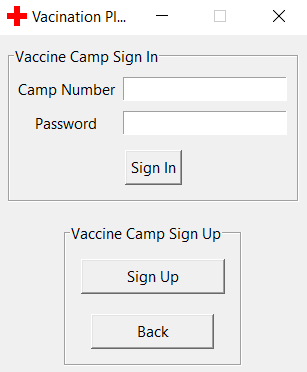
* Then the vp() function is called which changes the database as,

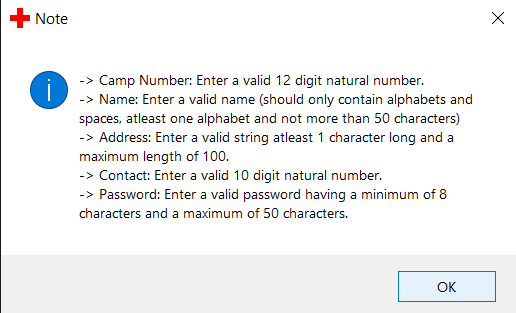
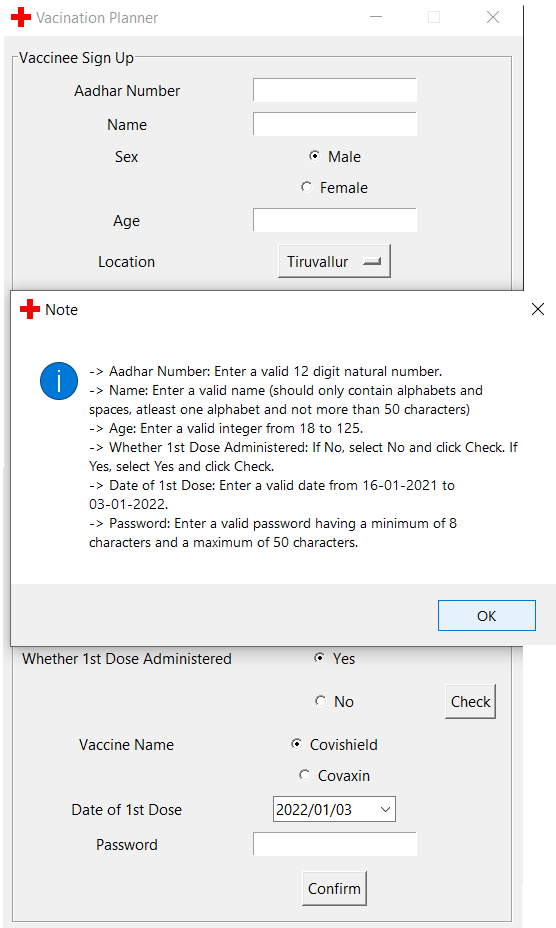
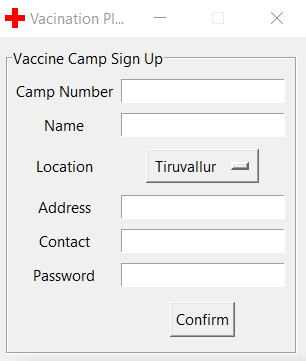
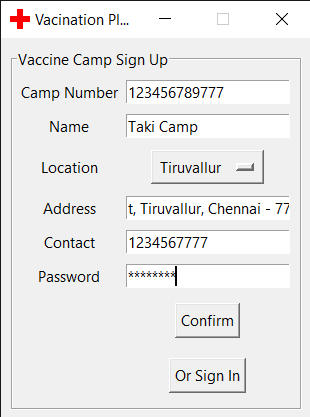
****

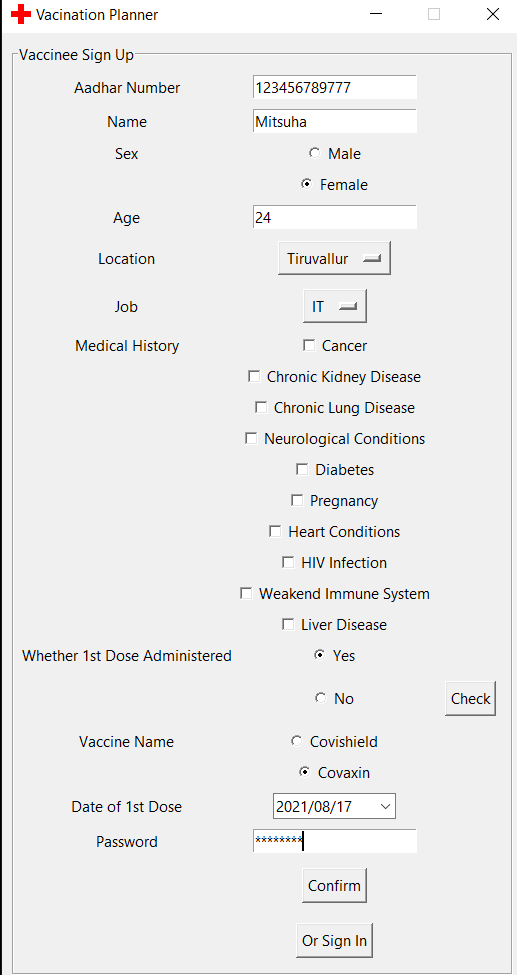
* **Steps for a vaccinee to sign up, then sign in and then finally check vaccine availability or update vaccine stock,**
  1. Click the button according to which user you are in the startup window, which shows Fig 1.
  2. Then click ‘Sign Up’, which then shows Fig 2.
  3. Read the given instructions and click ‘OK’, which then shows Fig 3.
  4. Fill up your details according to the instructions given and the click ‘Confirm’, which then shows Fig 4.
  5. Click ‘OK’, which takes you back to Fig 1.
  6. Now enter your details into sign in page and click ‘Sign in’, which if correct shows you Fig 5.
  7. Now as a vaccinee to check whether you are eligible for a vaccine click ‘Check for Vaccine’, which then shows your eligibility in Fig 6.
  8. Now for a vaccine camp user to enter the available vaccine stock click ‘Update Vaccine Stock’, which takes you to Fig 6.
     + Now read the instructions in Fig 6 and then click ‘OK’, which takes you to Fig 7.
     + Enter the details correctly and click ‘Update Stock’, which shows you Fig 8.
     + Click ‘OK’, which takes you back to Fig 5.
     + You can click ‘Update Vaccine Stock’ again to check the values you entered.

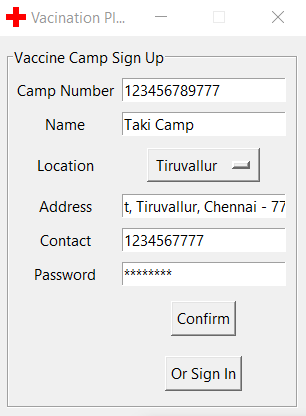
**Note:** You cannot update as a vaccinee after you receive a vaccine through this app. Otherwise you can update your details which is similar to the sign up process. Check Vaccine Camp Fig 5.

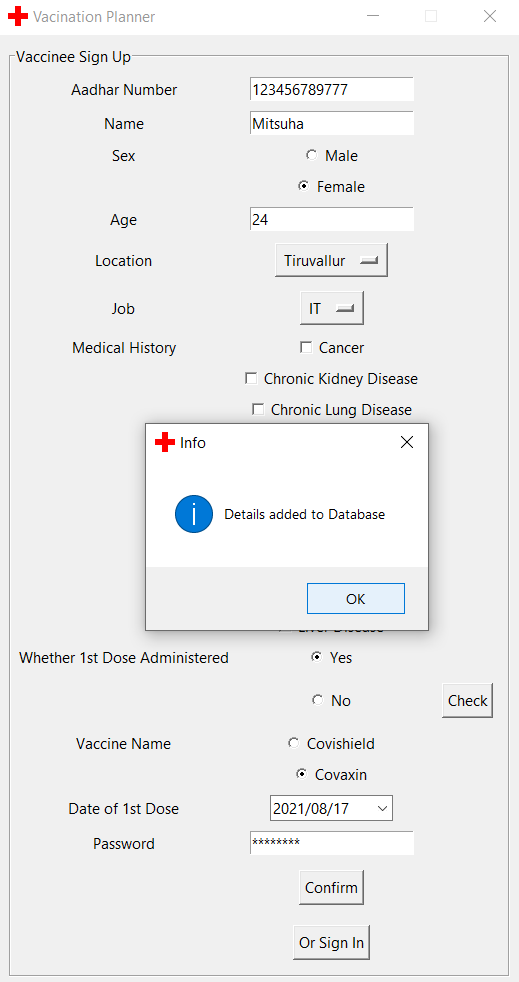
* After Step 5, the MySQL database is changed as shown in Fig 9.1 (vaccinee) and Fig 9.2 (vaccine camp).
* After Step 8.3, the MySQL database is changed as shown in Fig 10.
* **Fig 1**

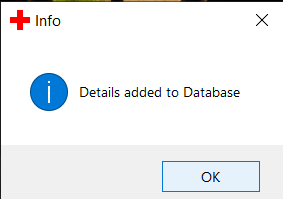
****

* **Fig 2**
* **Fig 3**

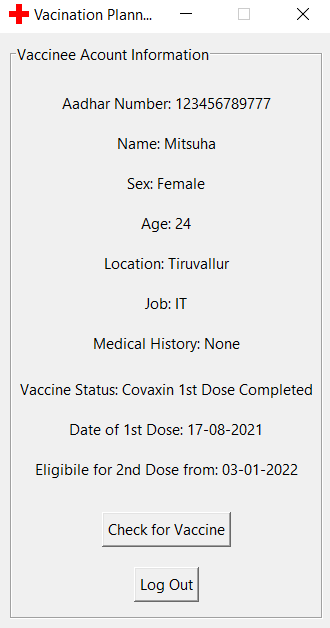
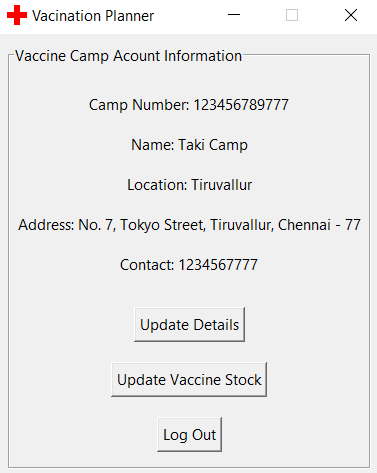


* **Fig 4**

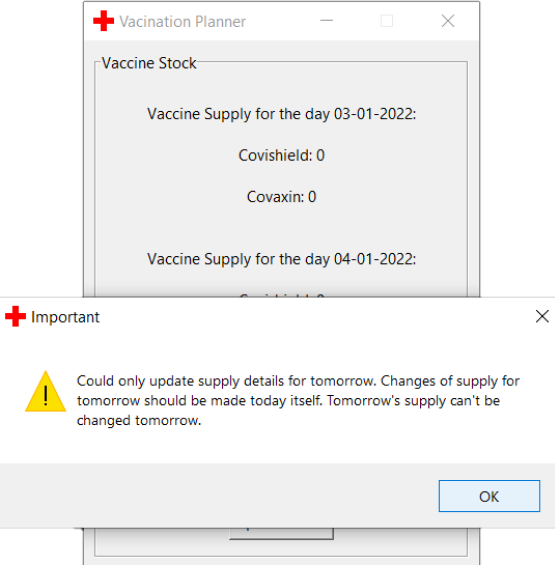
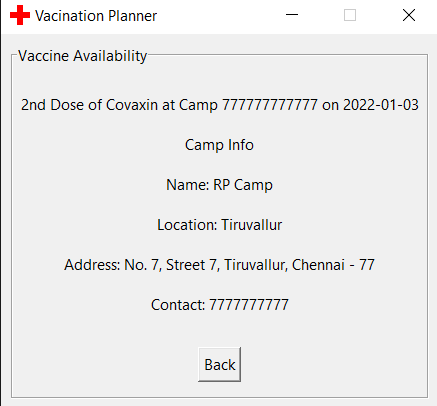
****

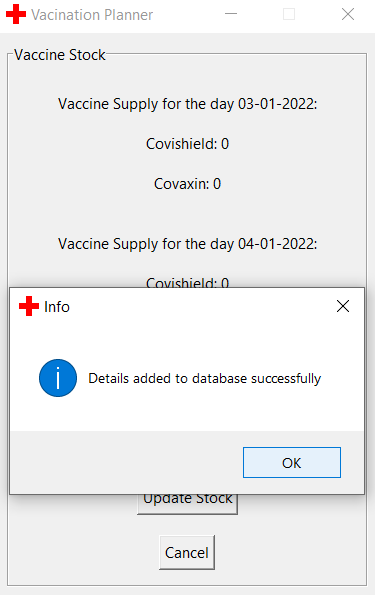
****

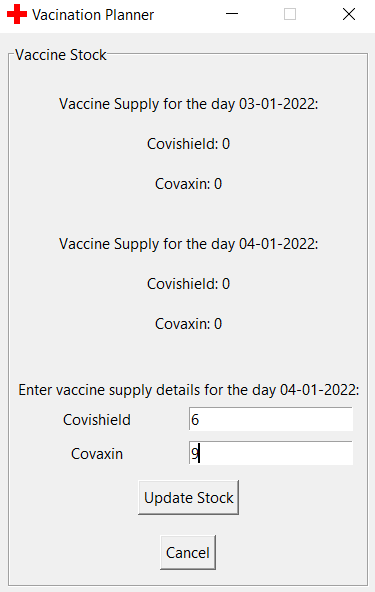
* **Fig 5**

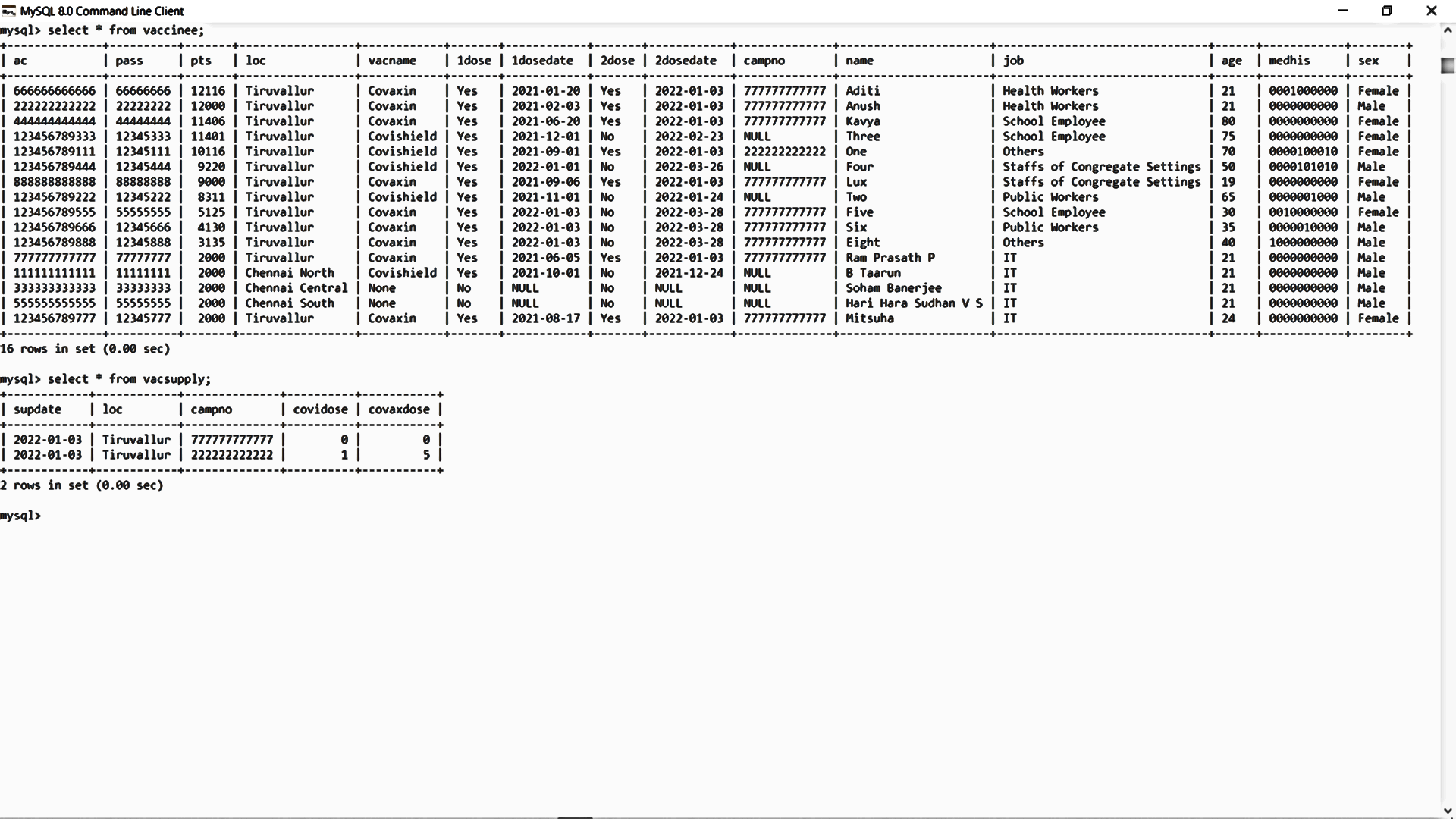
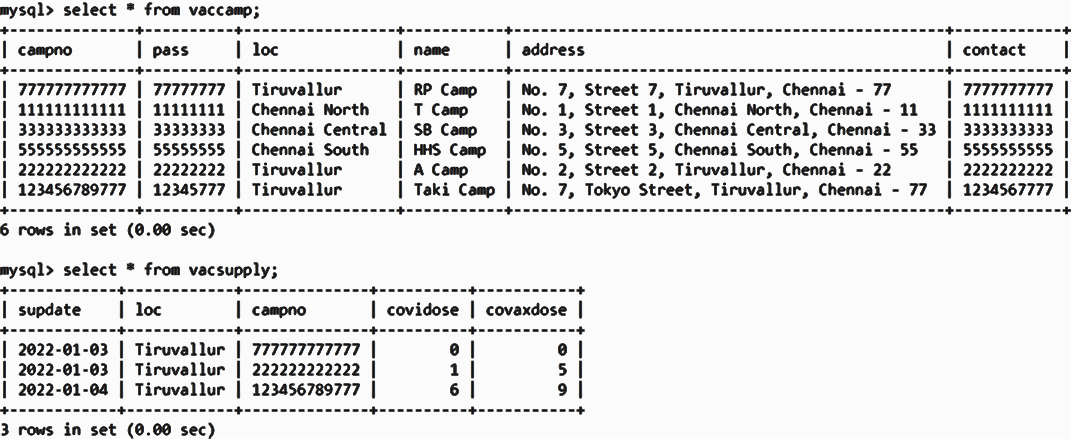
****

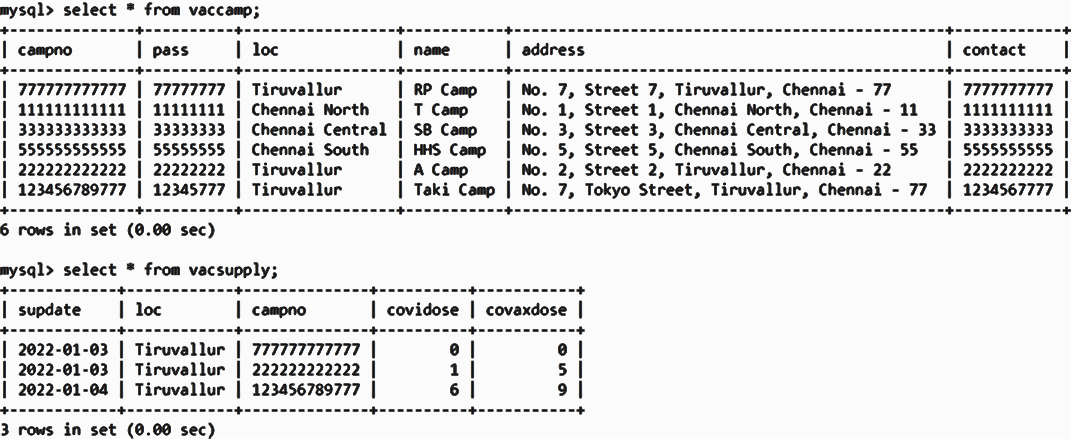
* **Fig 6**

****

* **Fig 7 ● Fig 8**

****

* **Fig 9.1**
* **Fig 9.2**
* **Fig 10**

****

**Bibliography**

* Computer Science with Python [Textbook XI] by Sumita Arora
* Computer Science with Python [Textbook XII] by Sumita Arora
* https://youtube.com/playlist?list=PLCC34OHNcOtoC6GglhF3ncJ5rLwQrLGnV
* https://www.health.nd.gov/covid-19-vaccine-priority-groups