VACCINE BOOKING SYSTEM

21CSS101J – PROGRAMMING FOR PROBLEM SOLVING

Mini Project Report

Submitted by

Tharun Subramanian.C [Reg. No.: RA2211003011187]
B.Tech. CSE - Core

SCHOOL OF COMPUTING
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
(Under Section 3 of UGC Act, 1956)
S.R.M. NAGAR, KATTANKULATHUR – 603 203
KANCHEEPURAM DISTRICT
December 2022

TABLE OF CONTENTS

PROBLEM STATEMENT

To implement a basic vaccine booking system using Python or C, allowing the users to easily book vaccine slots in their particular city according to their convenient date and time. The user can even book vaccine slots for their family members using their own account. In recent times, the lives of people have taken a serious turn. Lockdowns and restrictions everywhere we go. We have norms that we have never thought of before. Vaccines are one way we can protect ourselves from all these happening around the world. This project will make things simpler for people to book their vaccines making the world a safer place. Not long ago, most of the things we did were not online which made things a lot harder and confusing. A lot of mistakes were made and a lot of time was wasted. But these days it is neither safe for people to go out and stand in a queue nor do people have time for that.

METHODOLOGY/PROCEDURE

- This project consists of three main functions. The login function is used for the user to either register a new account using an username and a password or login into his/her database by using the already registered username and password.
- The next function is the registor function where the user will be asked to enter all his details such as details of the patient such as name, contact, email id, phone number, gender, city, and state.
- The final function used is the registraionform which creates the whole UI for this project which makes it simple for the user to use.
- This project makes it simpler for people to book vaccines and store a large amount of data and organize it.

Functions Used

- ✓ Login():

  This is used to log in or create a new account

- ✓ Registor():

  This is used to accept the details of the patient such as name, contact, email id, phone number, gender, city, and state

- ✓ Registrationform():

  This is used to create the UI for the whole project

SOURCE CODE

LOGIN CODE

```
from tkinter import *
from tkinter import messagebox
import mysql.connector

status = False
r_status = False
register_state = True
register_state_1 = True
user_id = ''
warn = 0
time_count = 500

def login_using_user(password):
    global user_id, status, register_state_1

    register_state_1 = True
    status = False

    # Creating and executing the login interface
    def login_interface():
        global register_state_1
        register_state_1 = False

        # Deleting the login or register screen
        try:
            screen.destroy()
        except:
            pass

        # Function for checking the password
        def get_password():
            global status, user_id, warn

            # Getting the user id and password
            user_id = user_entry.get().lower()
            password = password_entry.get()
            warning_count = warn

            # Checking the user id and password
            if user_id not in records:

                if warning_count in (0, 1):

                    warning_count += 1
                    warn = warning_count
                    available = 3 - warning_count
                    user_entry.delete(0, END)
                    password_entry.delete(0, END)
                    messagebox.showwarning('WARNING', 'USER not found')

                elif warning_count == 2:

                    # Displaying and ending the login screen
                    warning_count = 3
                    warn = warning_count
                    messagebox.showerror('ERROR', '3 WRONG ATTEMPTS')
                    root.destroy()
                    status = False

            elif records[user_id] != password:

                if warning_count in (0, 1):
                    # Clearing the entry field
                    user_entry.delete(0, END)
                    password_entry.delete(0, END)

                    # Displaying the error
                    warning_count += 1
                    warn = warning_count
                    available = 3 - warning_count
                    messagebox.showwarning('WARNING', 'PASSWORD does not match')

                elif warning_count == 2:

                    # Displaying and ending the login screen
                    warning_count = 3
                    warn = warning_count
                    messagebox.showerror('ERROR', '3 WRONG ATTEMPTS')
                    root.destroy()
                    status = False

            else:
                user_entry.delete(0, END)
                password_entry.delete(0, END)

                # Displaying the message and ending the screen
                messagebox.showinfo('SUCCESS', 'SUCCESSFULLY LOGGED IN')
                root.destroy()

                status = True

        # Initializing the screen
        root = Tk()
        root.title('LOGIN')

        # Getting the records
        database = mysql.connector.connect(host='localhost', user='root',
password='tharun', database='vaccine_project')
        cursor = database.cursor()
        cursor.execute('select * from passwords;')
        result = cursor.fetchall()
```

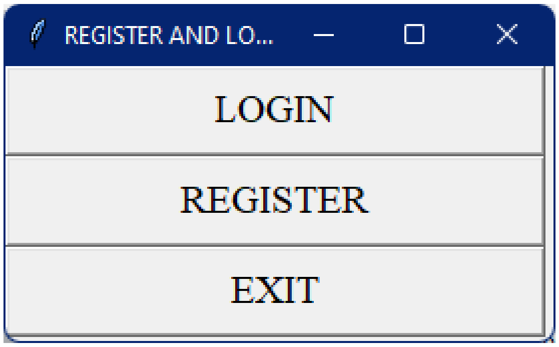
```
    records = {}
    for i in result:
        records.update({i[0]: i[1]})

    # Labels and buttons
    u_label = Label(root, text='User')
    u_label.grid(row=0, column=1)

    user_entry = Entry(root, width=30)
    user_entry.grid(row=0, column=2)

    p_label = Label(root, text='Password')
    p_label.grid(row=1, column=1)

    password_entry = Entry(root, show='*', width=30)
    password_entry.grid(row=1, column=2)

    login = Button(root, text='LOGIN', command=get_password)
    login.grid(row=2, column=1, columnspan=2)

    # Initiating the loop
    root.mainloop()

    if not status and warn < 3:
        login_using_user(password)

# Creating and executing the register interface
def register_interface():
    global register_state_1
    register_state_1 = False

    # Destroying the existing login or register screen
    screen.destroy()

    # Defining a function to register a record and add it to the database
    def insert_records():
        global user_id, status, r_status

        # Getting the user id and password
        user_id = user_entry.get().lower()
        password = password_entry.get()

        # Checking the existing records
        if password == '' or user_id == '':
            user_entry.delete(0, END)
            password_entry.delete(0, END)
            messagebox.showwarning('WARNING', 'USER or PASSWORD cannot be empty')

        elif user_id not in records:
            records[user_id] = password
            r_status = True

            # Adding the user to the database
```

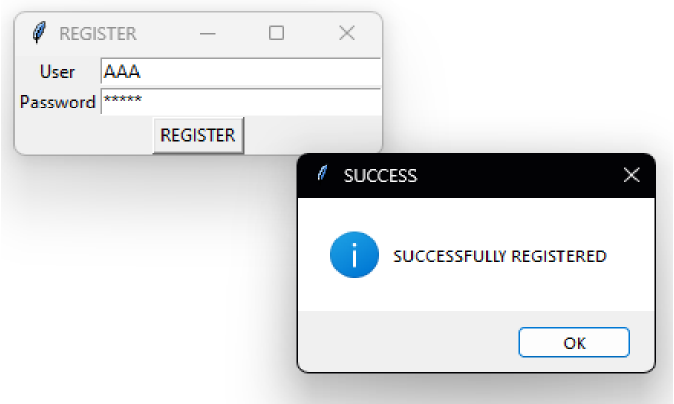
```
                cursor.execute('Insert into passwords values(\'%s\',\'%s\')' %
(user_id, password))
                database.commit()

                # Showing the success message and proceeding to log in to interface
                messagebox.showinfo("SUCCESS", "SUCCESSFULLY REGISTERED")
                root.destroy()

                login_interface()

            # Displaying error and getting a valid user and password
            else:
                user_entry.delete(0, END)
                password_entry.delete(0, END)
                messagebox.showwarning('WARNING', 'USER already exists')

        # Register interface
        root = Tk()
        root.title('REGISTER')

        # Getting the records
        database = mysql.connector.connect(host='localhost', user='root',
password='tharun', database='vaccine_project')
        cursor = database.cursor()
        cursor.execute('select * from passwords;')
        result = cursor.fetchall()
        records = {}
        for i in result:
            records.update({i[0]: i[1]})

        # Labels and buttons
        u_label = Label(root, text='User')
        u_label.grid(row=0, column=1)

        user_entry = Entry(root, width=30)
        user_entry.grid(row=0, column=2)

        p_label = Label(root, text='Password')
        p_label.grid(row=1, column=1)

        password_entry = Entry(root, show='*', width=30)
        password_entry.grid(row=1, column=2)

        login = Button(root, text='REGISTER', command=insert_records)
        login.grid(row=2, column=1, columnspan=2)

        root.mainloop()

        if not r_status:
            login_using_user(password)

    def exit_button():
        choice = messagebox.askyesno("EXIT", "Do you want to exit???")
        if choice:
            screen.destroy()

    # Database
    new_database = mysql.connector.connect(host='localhost', user='root',
password=password)
    new_cursor = new_database.cursor()
    new_cursor.execute('create database if not exists vaccine_project')
    new_cursor.execute('use vaccine_project')
    new_cursor.execute('create table if not exists passwords(user_id varchar(20),
password varchar(20))')

    # Login or Register screen
    screen = Tk()
    screen.title('REGISTER AND LOGIN')

    login_button = Button(screen, text='LOGIN', padx=97, pady=4,
command=login_interface, state=NORMAL, font=('Times', 15))
    login_button.grid(row=0, column=0, sticky=W+E)

    register_button = Button(screen, text='REGISTER', padx=61, pady=4,
command=register_interface, state=NORMAL, font=('Times', 15))
    register_button.grid(row=1, column=0, sticky=W+E)

    exit_button_display = Button(screen, text='EXIT', padx=100, pady=4,
command=exit_button, state=NORMAL, font=('Times', 15))
    exit_button_display.grid(row=2, column=0, sticky=W+E)

    screen.protocol('WM_DELETE_WINDOW', exit_button)

    screen.mainloop()

    return user_id, status
```

Main Code

```
from tkinter import *
from tkinter import ttk

# Importing connection
import mysql.connector
conn = mysql.connector.connect(user='root', password='tharun', host='localhost',
database='vaccine_project')

def registration_form():
    patient_list.destroy()
    def register():
        name1 = name.get()
        con1 = contact.get()
        email1 = email.get()
        gen1 = gender.get()
        city1 = citychoosen.get()
        hospital1 = hospchoosen.get()
        date1 = datechoosen.get()
        slot1 = slotchoosen.get()
        vaccine1 = vaccine.get()
        hospital1 = hospital.replace(' ', '_')


        if name1 == '' or con1 == '' or email1 == '' or gen1 == 0 or city1.upper() ==
'NONE' or hospital1 == 'NONE' or date1.upper() == 'NONE' or slot1.upper() == 'NONE' or
vaccine1 == 0:
            message.set("Fill all the fields")

        elif len(con1) == 0 or not con1.isdigit() or len(con1)!=10 :
            message.set("Provide valid contact number")

        elif len(email1.split('@')) != 2 or email1[-4:] != '.com'or
len(email1.split('@')[1].split('.')[0]) == 0:
            message.set("Provide valid email")

        else:
            cursor = conn.cursor()
            insert_stmt = "INSERT INTO patient_list(NAME, CONTACT, EMAIL, GENDER, CITY,
USER_ID, VACCINE, dov, slot_no, hospital_name) VALUES (%s, %s, %s, %s, %s, %s, %s, %s,
%s, %s)"

            if gen1 == 1:
                gen_name = 'Male'
            else:
                gen_name = 'Female'

            if vaccine1 == 1:
                vac_name = 'Covaccine'
            else:
                vac_name = 'Covishield'
            slot_no = ['8:30 - 9:00', '9:00 - 9:30', '9:30 - 10:00', '10:00 -
10:30'].index(slot1) + 1
            data = (name1, con1, email1, gen_name, city1, user_id, vac_name, date1,
slot1, hospital1)

            cursor.execute(insert_stmt, data)
            cursor.execute('update %s set slot%d = slot%d - 1 where dov = "%s"' %
(hospital1, slot_no, slot_no, date1))
            conn.commit()
            # conn.rollback()

            message.set("Stored successfully")

            reg_screen.destroy()



    def alter_tree(event):
        global city

        city = citychoosen.get()
        citychoosen.set(city)
        req_list = []
        cursor = conn.cursor()
        cursor.execute('select name from hospital where city = "%s"' % city)
        req_list = cursor.fetchall()
        req_list = [i[0] for i in req_list]

        hospchoosen['values'] = req_list

        hospchoosen.set('NONE')
        datechoosen.set('SELECT HOSPITAL')
        datechoosen['values'] = []
        slotchoosen.set('SELECT DATE')
        slotchoosen['values'] = []

    def alter_tree_1(event):
        global hospital

        hospital = hospchoosen.get()
        hospchoosen.set(hospital)
        hospital1 = hospital.replace(' ', '_')
        req_list = []
        cursor = conn.cursor()
        cursor.execute('create table if not exists %s(dov date primary key, slot1 int
default 5, slot2 int default 5, slot3 int default 5, slot4 int default 5)' % hospital1)
        cursor.execute('delete from %s where dov < date(now())' % hospital1)
        for i in range(7):
            try:
                cursor.execute('insert into %s(dov) values((SELECT
DATE_ADD(date(now()), INTERVAL %d DAY)));' % (hospital1, i))
            except:
                pass
        conn.commit()
        cursor.execute('select dov from %s where slot1 > 0 or slot2 > 0 or slot3 > 0 or
slot4 > 0' % hospital1)
        req_list = cursor.fetchall()
        req_list = [i[0] for i in req_list]

        datechoosen['values'] = req_list

        datechoosen.set('NONE')
        slotchoosen.set('SELECT DATE')
        slotchoosen['values'] = []

    def alter_tree_2(event):
        global hospital, slot, date

        hospital1 = hospital.replace(' ', '_')
        date = datechoosen.get()
        datechoosen.set(date)
        req_list = []
        cursor = conn.cursor()
        cursor.execute('select slot1, slot2, slot3, slot4 from %s where dov = "%s"' %
(hospital1, date))
        req_list = cursor.fetchall()[0]
        slot1, slot2, slot3, slot4 = req_list
        req_list = []
        if slot1 > 0:
            req_list.append('8:30 - 9:00')

        if slot2 > 0:
            req_list.append('9:00 - 9:30')

        if slot3 > 0:
            req_list.append('9:30 - 10:00')

        if slot4 > 0:
            req_list.append('10:00 - 10:30')

        slotchoosen['values'] = req_list

        slotchoosen.set('NONE')

    global reg_screen
    reg_screen = Tk()
    reg_screen.title("Registration Form")

    # Setting height and width of screen
    reg_screen.geometry("350x480")
    global message
    global name

    global email
    global gender
    global city
    global hospital
    global date
    global slot
    global vaccine

    name = StringVar()
    contact = StringVar()
    email = StringVar()
    gender = IntVar()
    vaccine = IntVar()
    city = StringVar()
    state = StringVar()
    message = StringVar()
    hospital = StringVar()
    date = StringVar()
    slot = IntVar()

    Label(reg_screen, width="300", text="Please enter details below", bg="lightblue",
fg="red").pack()

    # Name Label
    Label(reg_screen, text="Name * ").place(x=20, y=40)

    # Name textbox
    Entry(reg_screen, textvariable=name).place(x=90, y=44)

    # Contact Label
    Label(reg_screen, text="Contact * ").place(x=20, y=80)

    # Contact textbox
    Entry(reg_screen, textvariable=contact).place(x=90, y=80)

    # email Label
    Label(reg_screen, text="Email * ").place(x=20, y=120)

    # email textbox
    Entry(reg_screen, textvariable=email).place(x=90, y=122)

    # gender Label
    Label(reg_screen, text="Gender * ").place(x=20, y=160)

    # gender radiobutton
    Radiobutton(reg_screen, text="Male", variable=gender, value=1).place(x=90, y=162)
    Radiobutton(reg_screen, text="Female", variable=gender, value=2).place(x=150,
y=162)

    Label(reg_screen, text="Vaccine * ").place(x=20, y=202)

    Radiobutton(reg_screen, text="Covaccine", variable=vaccine, value=1).place(x=90,
y=202)
    Radiobutton(reg_screen, text="Covishield", variable=vaccine, value=2).place(x=170,
y=202)

    # city Label
    Label(reg_screen, text="City * ").place(x=20, y=242)
    # city combobox
    citychoosen = ttk.Combobox(reg_screen, width=27, textvariable=city)
    citychoosen['values'] = ('NONE', 'Chennai', 'Mumbai', 'Bangalore', 'Kochi',
'Kolkata',)
    citychoosen.current(0)
    citychoosen.place(x=90, y=242)
    citychoosen.bind('<<ComboboxSelected>>', alter_tree)

    # hosp Label
    Label(reg_screen, text="Hospital * ").place(x=20, y=282)
    # hosp combobox
    hospchoosen = ttk.Combobox(reg_screen, width=27, textvariable=hospital)
    hospchoosen['values'] = []
    hospchoosen.set('SELECT HOSPITAL')
    hospchoosen.place(x=90, y=282)
    hospchoosen.bind('<<ComboboxSelected>>', alter_tree_1)

    # date Label
    Label(reg_screen, text="Date * ").place(x=20, y=322)
    # date combobox
    datechoosen = ttk.Combobox(reg_screen, width=27, textvariable=date)
    datechoosen['values'] = []
    datechoosen.set('SELECT DATE')
    datechoosen.place(x=90, y=322)
    datechoosen.bind('<<ComboboxSelected>>', alter_tree_2)

    # slot Label
    Label(reg_screen, text="Slot * ").place(x=20, y=362)
    # slot combobox
    slotchoosen = ttk.Combobox(reg_screen, width=27, textvariable=slot)
    slotchoosen['values'] = []
    slotchoosen.set('SELECT SLOT')
    slotchoosen.place(x=90, y=362)
    slotchoosen.bind('<<ComboboxSelected>>')

    # Label for displaying login status[success/failed]

    Label(reg_screen, text="", textvariable=message).place(x=95, y=442)
    # Login button
    Button(reg_screen, text="Register", width=10, height=1, bg="gold",
command=register).place(x=105, y=402)

    reg_screen.mainloop()



def vacancy():
    # city Label
    Label(reg_screen, text="City * ").place(x=20, y=242)
    # city combobox
    citychoosen = ttk.Combobox(reg_screen, width=27, textvariable=city)
    citychoosen['values'] = ('NONE', 'Chennai', 'Mumbai', 'Bangalore', 'Kochi',
'Kolkata',)
    citychoosen.current(0)
    citychoosen.place(x=90, y=242)
    citychoosen.bind('<<ComboboxSelected>>', alter_tree)
    # hosp Label
    Label(reg_screen, text="Hospital * ").place(x=20, y=282)
    # hosp combobox
    hospchoosen = ttk.Combobox(reg_screen, width=27, textvariable=hospital)
    hospchoosen['values'] = []
    hospchoosen.set('SELECT CITY')
    hospchoosen.place(x=90, y=282)
    hospchoosen.bind('<<ComboboxSelected>>', alter_tree_1)



def form():
    patient_list.destroy()

    cursor = conn.cursor()
    cursor.execute('select Name, Contact, Email, Gender, City, Vaccine, Dov, Slot_no,
Hospital_Name from patient_list where user_id = "%s"' % user_id)
    result = cursor.fetchall()

    if not result:
        result = [('', '', '', 'NO', 'RECORDS', 'FOUND', '', '', '')]

    def back_function():
        display_screen.destroy()

    display_screen = Tk()
    display_screen.title('PATIENT LIST')

    user_label = Label(display_screen, text='USER_ID : '+user_id, font=('Times New
roman', 12))
    user_label.grid(row=0, column=0)

    patient_frame = Frame(display_screen)
    patient_frame.grid(row=1, column=0)

    back = Button(display_screen, text='BACK', command=back_function)
    back.grid(row=2, column=0, sticky=N + S + W + E, pady=4)

    scroll = Scrollbar(patient_frame, orient=VERTICAL)
    scroll.pack(side=RIGHT, fill='y')

    high_score = ttk.Treeview(patient_frame, height=7, yscrollcommand=scroll.set)
    high_score.pack()

    scroll.config(command=high_score.yview)

    high_score['columns'] = ("Name", "Contact", "Email", "Gender", "City", "Vaccine",
"Date", "Slot", "Hospital")

    high_score.column("#0", width=0, stretch=NO)
    high_score.column("Name", anchor=CENTER, width=150)
    high_score.column("Contact", anchor=CENTER, width=100)
    high_score.column("Email", anchor=CENTER, width=150)
    high_score.column("Gender", anchor=CENTER, width=70)
    high_score.column("City", anchor=CENTER, width=100)
    high_score.column("Vaccine", anchor=CENTER, width=80)
    high_score.column("Date", anchor=CENTER, width=80)
    high_score.column("Slot", anchor=CENTER, width=80)
    high_score.column("Hospital", anchor=CENTER, width=150)

    high_score.heading('Name', text='Name', anchor=CENTER)
    high_score.heading('Contact', text='Contact', anchor=CENTER)
    high_score.heading('Email', text='Email', anchor=CENTER)
    high_score.heading('Gender', text='Gender', anchor=CENTER)
    high_score.heading('City', text='City', anchor=CENTER)
    high_score.heading('Vaccine', text='Vaccine', anchor=CENTER)
    high_score.heading('Date', text='Date', anchor=CENTER)
    high_score.heading('Slot', text='Slot', anchor=CENTER)
    high_score.heading('Hospital', text='Hospital', anchor=CENTER)

    for position in range(len(result)):
        result[position] = list(result[position])
        high_score.insert(parent='', index=END, iid=position, text='',
value=result[position])

    display_screen.mainloop()


import login
user_id, status = login.login_using_user('tharun')

def exit_button():
    patient_list.destroy()

    global status
    status = False
    exit()



while status:
    patient_list = Tk()
    patient_list.geometry('250x250')
    patient_list.title('Main Window')

    userid = Label(patient_list, text='USER_ID : '+user_id, font=('Times New roman',
12))
    userid.pack(anchor=NE, padx=20, pady=8)
    newp = Button(patient_list, text='REGISTER A PATIENT', command=registration_form,
font=('Times New roman', 12))
    newp.pack(padx=20, pady=8)
    oldp = Button(patient_list, text='VIEW PATIENTS', command=form, font=('Times New
roman', 12))
    oldp.pack(padx=20, pady=8)

    exitb = Button(patient_list, text='EXIT', command=exit_button, font=('Times New
roman', 12))
    exitb.pack(padx=20, pady=8)

    patient_list.protocol('WM_DELETE_WINDOW', exit_button)
    patient_list.mainloop()
```

RESULTS

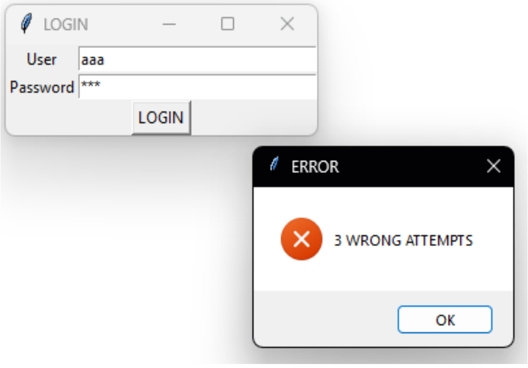




User AAA

Password *****

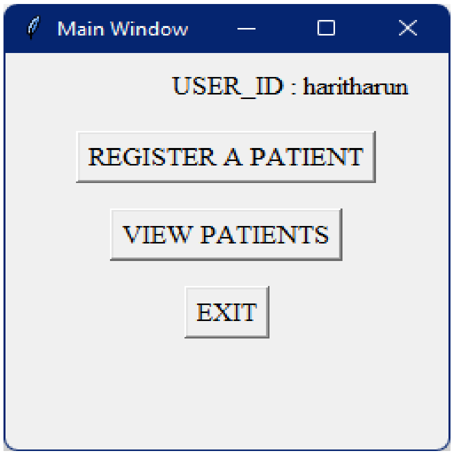
SUCCESSFULLY REGISTERED



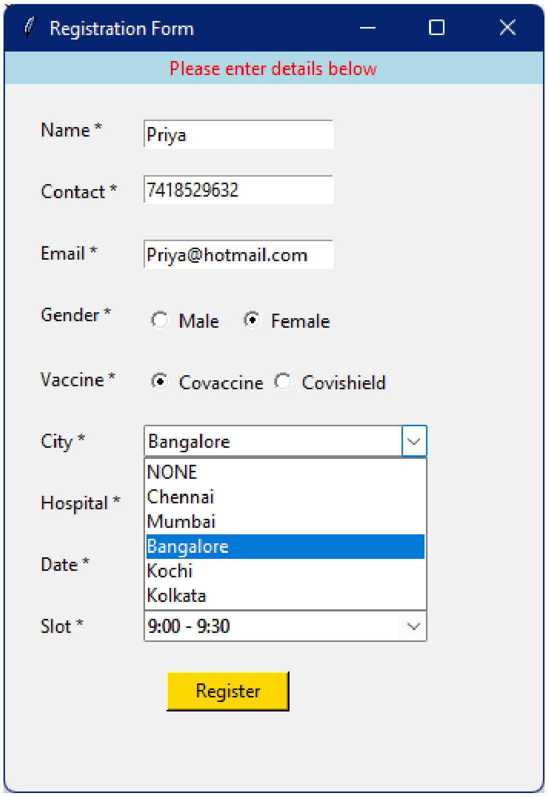
User aaa

Password ***

3 WRONG ATTEMPTS



USER_ID : haritharun



Registration Form

Please enter details below

Name * Priya

Contact * 7418529632

Email * Priya@hotmail.com

Gender * ○ Male ● Female

Vaccine * ● Covaccine ○ Covishield

City * Bangalore

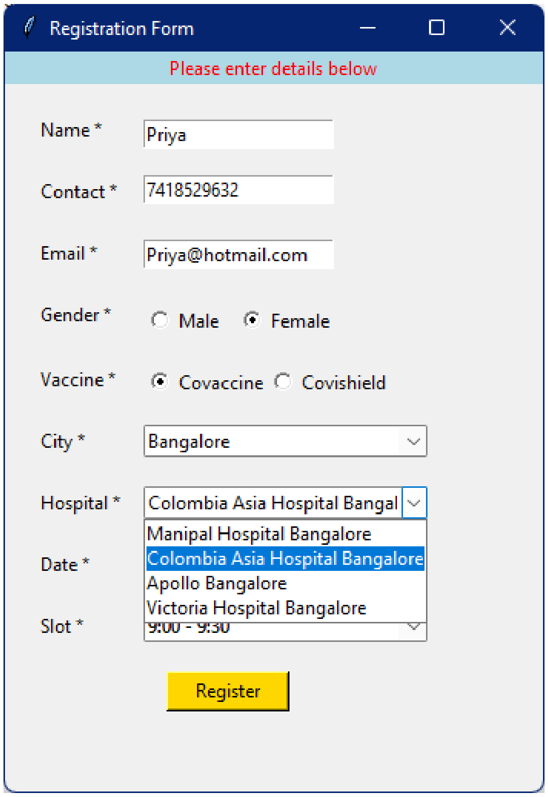
NONE
Chennai
Mumbai
Bangalore
Kochi
Kolkata

Hospital *

Date *

Slot * 9:00 - 9:30

Register



Registration Form

Please enter details below

Name * Priya

Contact * 7418529632

Email * Priya@hotmail.com

Gender * ○ Male ● Female

Vaccine * ● Covaccine ○ Covishield

City * Bangalore

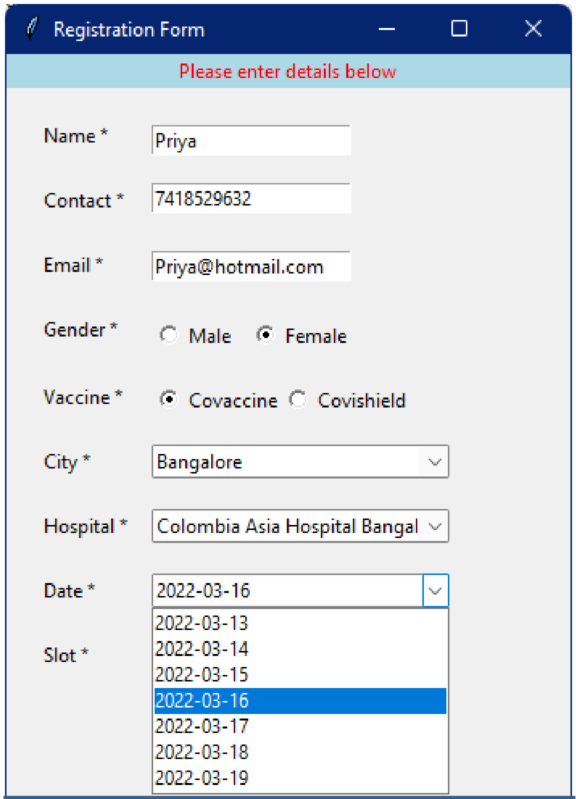
Hospital * Colombia Asia Hospital Bangal

Manipal Hospital Bangalore
Colombia Asia Hospital Bangalore
Apollo Bangalore
Victoria Hospital Bangalore

Date *

Slot * 9:00 - 9:30

Register



Registration Form

Please enter details below

Name * Priya

Contact * 7418529632

Email * Priya@hotmail.com

Gender * ○ Male ● Female

Vaccine * ● Covaccine ○ Covishield

City * Bangalore

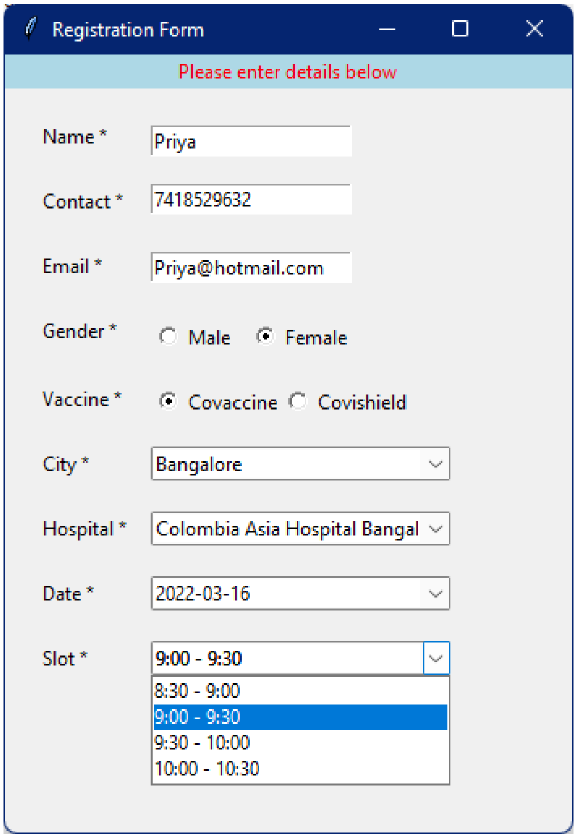
Hospital * Colombia Asia Hospital Bangal

Date * 2022-03-16

2022-03-13
2022-03-14
2022-03-15
2022-03-16
2022-03-17
2022-03-18
2022-03-19

Slot *

7 days from the present day is displayed



Registration Form

Please enter details below

Name * Priya

Contact * 7418529632

Email * Priya@hotmail.com

Gender * ○ Male ● Female

Vaccine * ● Covaccine ○ Covishield

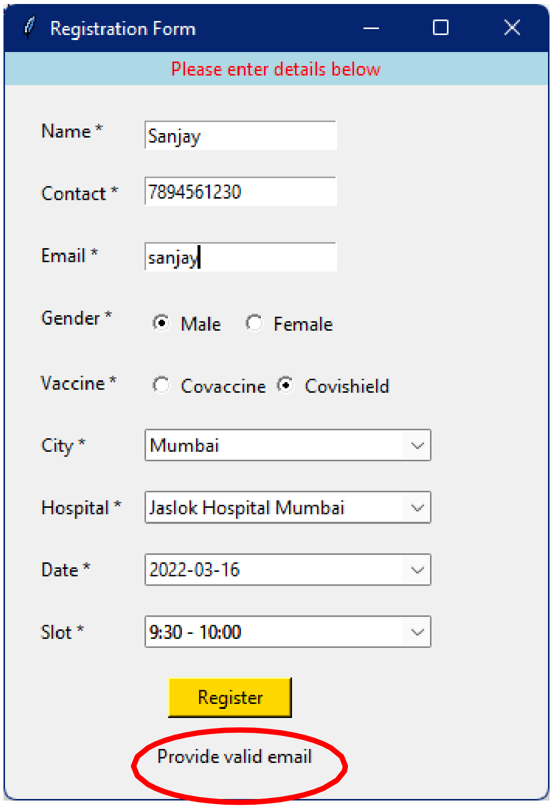
City * Bangalore

Hospital * Colombia Asia Hospital Bangal

Date * 2022-03-16

Slot * 9:00 - 9:30

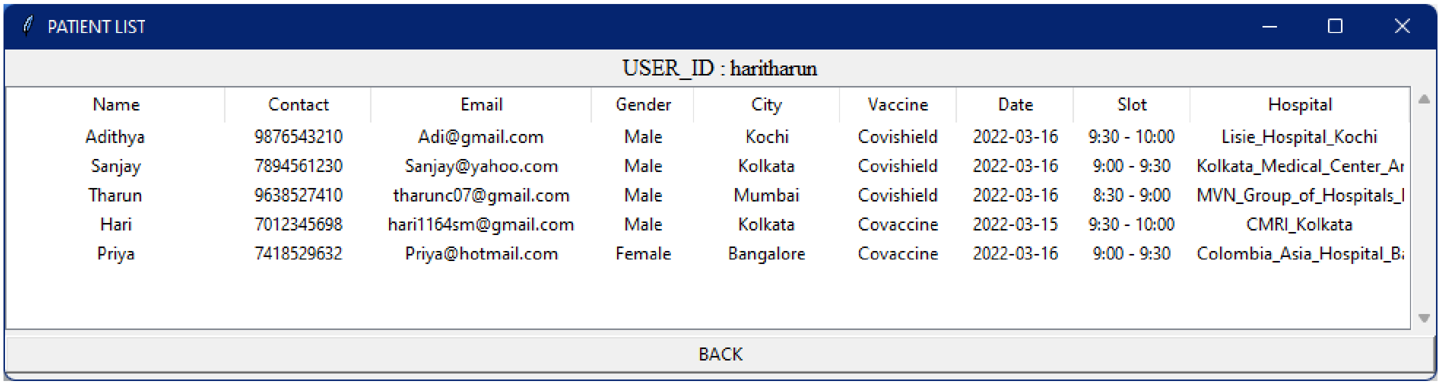
8:30 - 9:00
9:00 - 9:30
9:30 - 10:00
10:00 - 10:30



Please enter details below

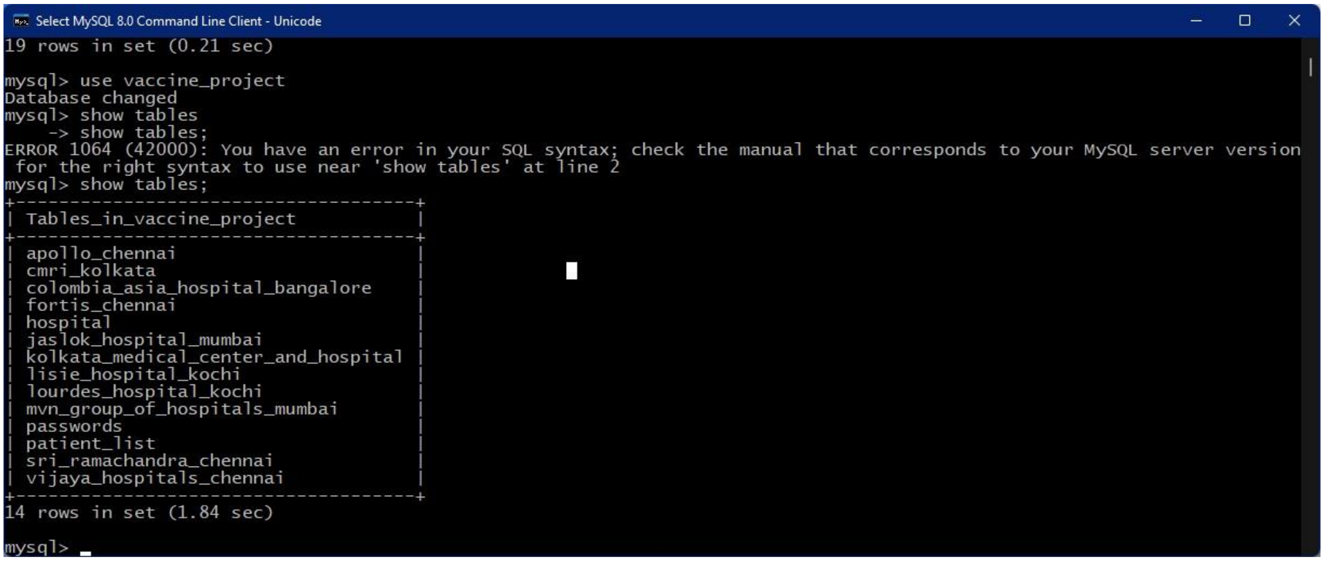
Name *	Sanjay
Contact *	7894561230
Email *	sanjay
Gender *	● Male ○ Female
Vaccine *	○ Covaccine ● Covishield
City *	Mumbai
Hospital *	Jaslok Hospital Mumbai
Date *	2022-03-16
Slot *	9:30 - 10:00

Provide valid email

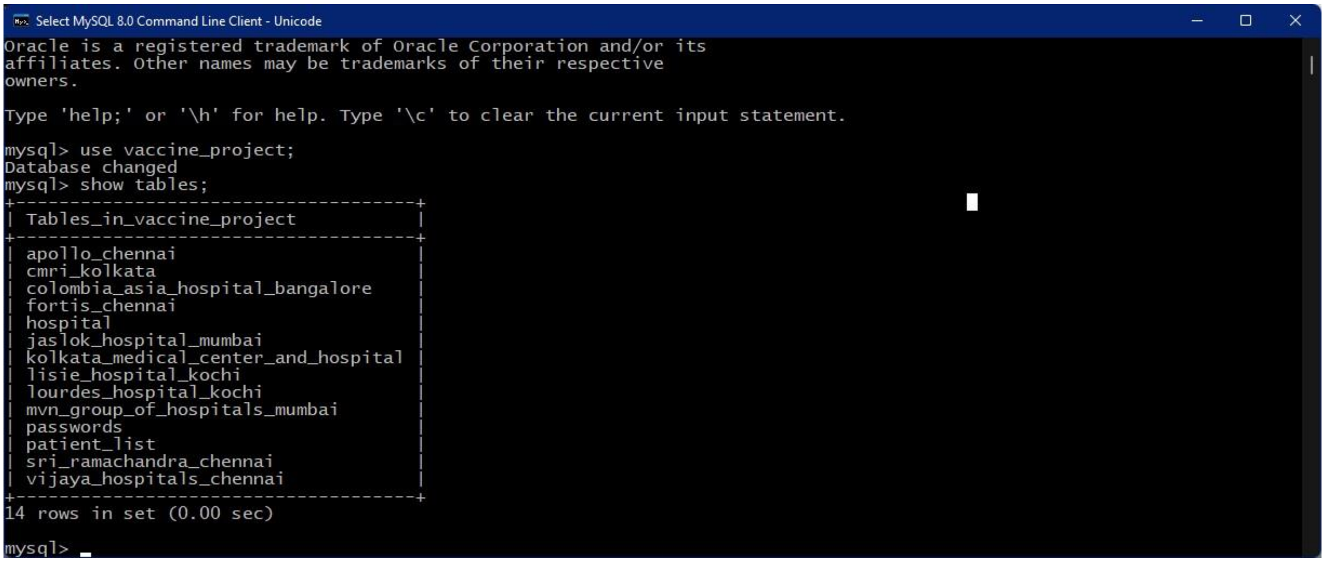


USER_ID : haritharun

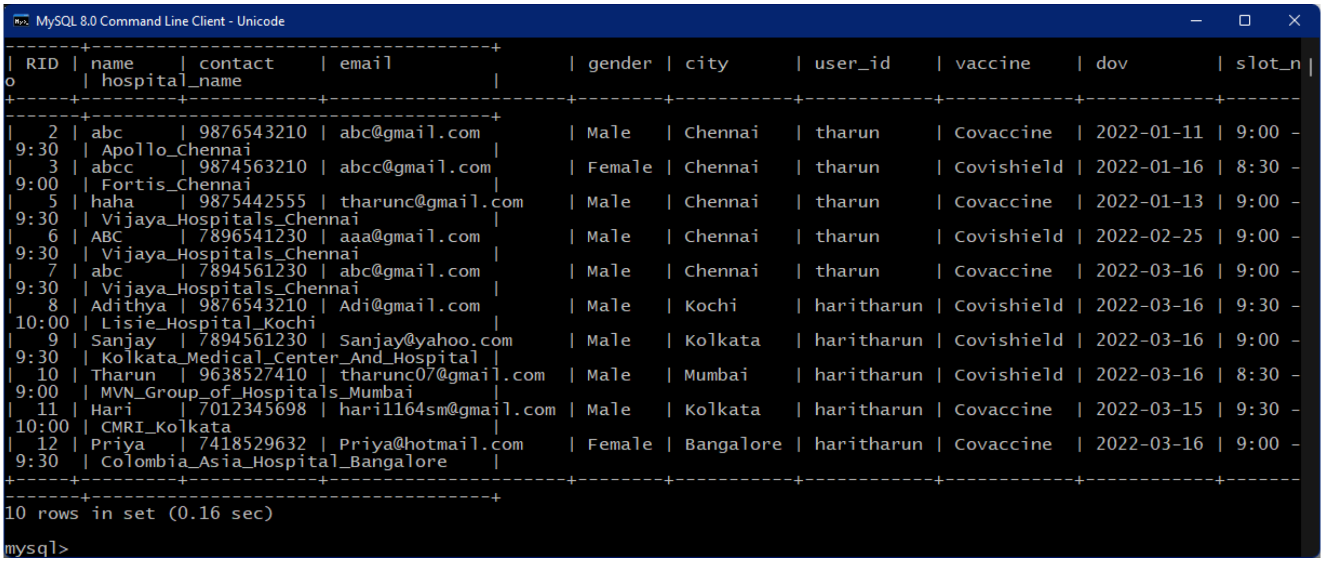
Name	Contact	Email	Gender	City	Vaccine	Date	Slot	Hospital
Adithya	9876543210	Adi@gmail.com	Male	Kochi	Covishield	2022-03-16	9:30 - 10:00	Lisie_Hospital_Kochi
Sanjay	7894561230	Sanjay@yahoo.com	Male	Kolkata	Covishield	2022-03-16	9:00 - 9:30	Kolkata_Medical_Center_Ar
Tharun	9638527410	tharunc07@gmail.com	Male	Mumbai	Covishield	2022-03-16	8:30 - 9:00	MVN_Group_of_Hospitals_I
Hari	7012345698	hari1164sm@gmail.com	Male	Kolkata	Covaccine	2022-03-15	9:30 - 10:00	CMRI_Kolkata
Priya	7418529632	Priya@hotmail.com	Female	Bangalore	Covaccine	2022-03-16	9:00 - 9:30	Colombia_Asia_Hospital_Ba



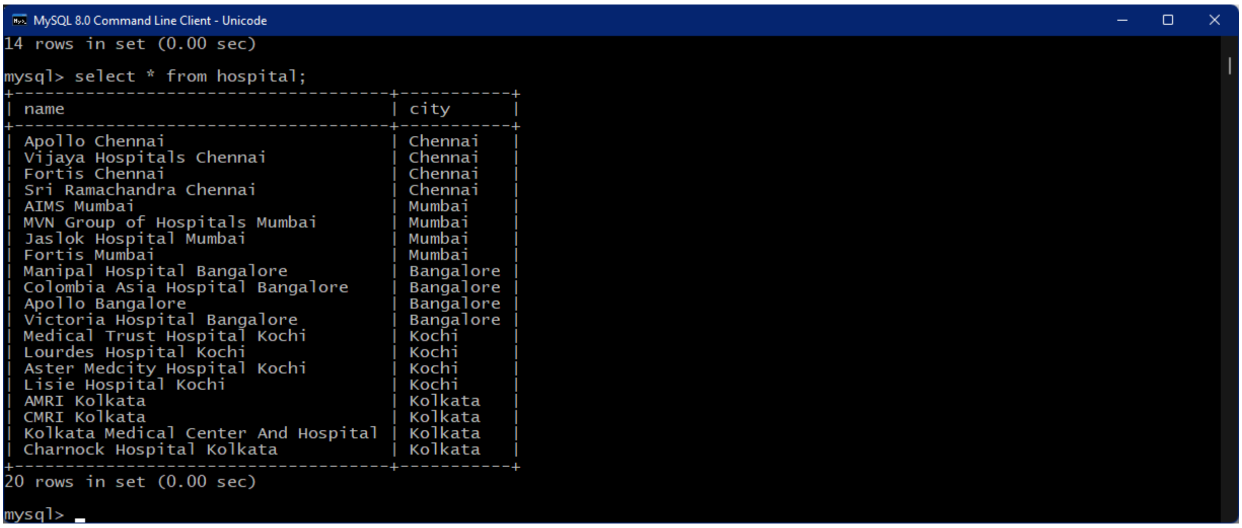
```
19 rows in set (0.21 sec)

mysql> use vaccine_project
Database changed
mysql> show tables
    -> show tables;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version
 for the right syntax to use near 'show tables' at line 2
mysql> show tables;
+-------------------------------------+
| Tables_in_vaccine_project           |
+-------------------------------------+
| apollo_chennai                      |
| cmri_kolkata                        |
| colombia_asia_hospital_bangalore    |
| fortis_chennai                      |
| hospital                            |
| jaslok_hospital_mumbai              |
| kolkata_medical_center_and_hospital |
| lisie_hospital_kochi                |
| lourdes_hospital_kochi              |
| mvn_group_of_hospitals_mumbai       |
| passwords                           |
| patient_list                        |
| sri_ramachandra_chennai             |
| vijaya_hospitals_chennai            |
+-------------------------------------+
14 rows in set (1.84 sec)

mysql> _
```



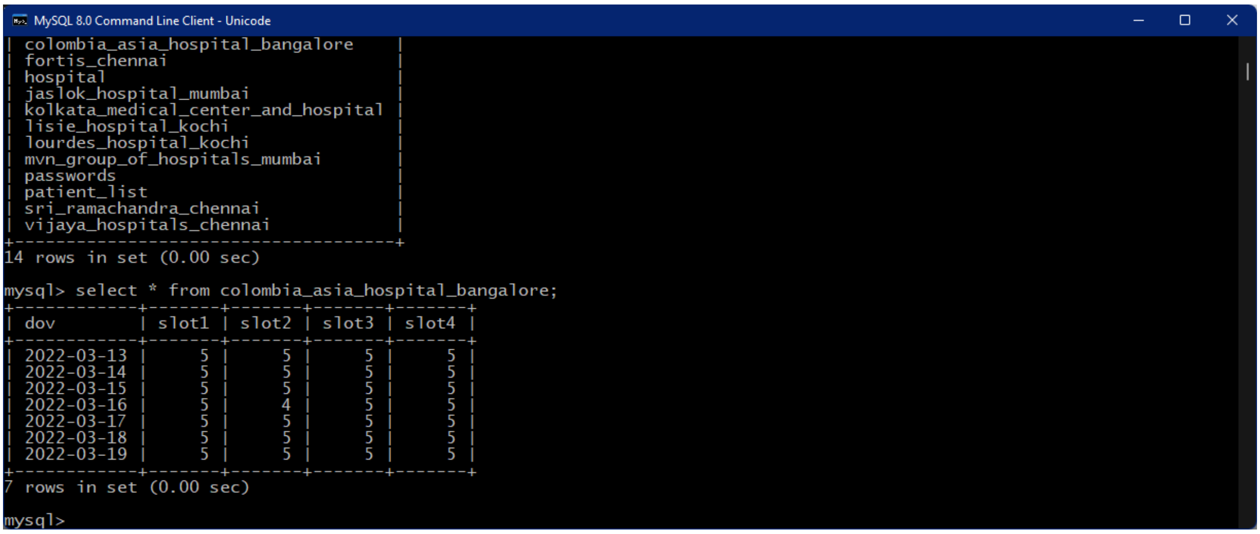
```
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use vaccine_project;
Database changed
mysql> show tables;
+-------------------------------------+
| Tables_in_vaccine_project           |
+-------------------------------------+
| apollo_chennai                      |
| cmri_kolkata                        |
| colombia_asia_hospital_bangalore    |
| fortis_chennai                      |
| hospital                            |
| jaslok_hospital_mumbai              |
| kolkata_medical_center_and_hospital |
| lisie_hospital_kochi                |
| lourdes_hospital_kochi              |
| mvn_group_of_hospitals_mumbai       |
| passwords                           |
| patient_list                        |
| sri_ramachandra_chennai             |
| vijaya_hospitals_chennai            |
+-------------------------------------+
14 rows in set (0.00 sec)

mysql> _
```



```
-------+-------------------------------------+
| RID | name    | contact    | email                | gender | city      | user_id    | vaccine    | dov        | slot_n
o      | hospital_name                       |
+-----+---------+------------+----------------------+--------+-----------+------------+------------+------------+-------
-------+-------------------------------------+
|   2 | abc     | 9876543210 | abc@gmail.com        | Male   | Chennai   | tharun     | Covaccine  | 2022-01-11 | 9:00 -
 9:30  | Apollo_Chennai                      |
|   3 | abcc    | 9874563210 | abcc@gmail.com       | Female | Chennai   | tharun     | Covishield | 2022-01-16 | 8:30 -
 9:00  | Fortis_Chennai                      |
|   5 | haha    | 9875442555 | tharunc@gmail.com    | Male   | Chennai   | tharun     | Covaccine  | 2022-01-13 | 9:00 -
 9:30  | Vijaya_Hospitals_Chennai            |
|   6 | ABC     | 7896541230 | aaa@gmail.com        | Male   | Chennai   | tharun     | Covishield | 2022-02-25 | 9:00 -
 9:30  | Vijaya_Hospitals_Chennai            |
|   7 | abc     | 7894561230 | abc@gmail.com        | Male   | Chennai   | tharun     | Covaccine  | 2022-03-16 | 9:00 -
 9:30  | Vijaya_Hospitals_Chennai            |
|   8 | Adithya | 9876543210 | Adi@gmail.com        | Male   | Kochi     | haritharun | Covishield | 2022-03-16 | 9:30 -
 10:00 | Lisie_Hospital_Kochi                |
|   9 | Sanjay  | 7894561230 | Sanjay@yahoo.com     | Male   | Kolkata   | haritharun | Covishield | 2022-03-16 | 9:00 -
 9:30  | Kolkata_Medical_Center_And_Hospital |
|  10 | Tharun  | 9638527410 | tharunc07@gmail.com  | Male   | Mumbai    | haritharun | Covishield | 2022-03-16 | 8:30 -
 9:00  | MVN_Group_of_Hospitals_Mumbai       |
|  11 | Hari    | 7012345698 | hari1164sm@gmail.com | Male   | Kolkata   | haritharun | Covaccine  | 2022-03-15 | 9:30 -
 10:00 | CMRI_Kolkata                        |
|  12 | Priya   | 7418529632 | Priya@hotmail.com    | Female | Bangalore | haritharun | Covaccine  | 2022-03-16 | 9:00 -
 9:30  | Colombia_Asia_Hospital_Bangalore    |
+-----+---------+------------+----------------------+--------+-----------+------------+------------+------------+-------
-------+-------------------------------------+
10 rows in set (0.16 sec)

mysql>
```



```
14 rows in set (0.00 sec)

mysql> select * from hospital;
+-------------------------------------+-----------+
| name                                | city      |
+-------------------------------------+-----------+
| Apollo Chennai                      | Chennai   |
| Vijaya Hospitals Chennai            | Chennai   |
| Fortis Chennai                      | Chennai   |
| Sri Ramachandra Chennai             | Chennai   |
| AIMS Mumbai                         | Mumbai    |
| MVN Group of Hospitals Mumbai       | Mumbai    |
| Jaslok Hospital Mumbai              | Mumbai    |
| Fortis Mumbai                       | Mumbai    |
| Manipal Hospital Bangalore          | Bangalore |
| Colombia Asia Hospital Bangalore    | Bangalore |
| Apollo Bangalore                    | Bangalore |
| Victoria Hospital Bangalore         | Bangalore |
| Medical Trust Hospital Kochi        | Kochi     |
| Lourdes Hospital Kochi              | Kochi     |
| Aster Medcity Hospital Kochi        | Kochi     |
| Lisie Hospital Kochi                | Kochi     |
| AMRI Kolkata                        | Kolkata   |
| CMRI Kolkata                        | Kolkata   |
| Kolkata Medical Center And Hospital | Kolkata   |
| Charnock Hospital Kolkata           | Kolkata   |
+-------------------------------------+-----------+
20 rows in set (0.00 sec)

mysql> _
```



```
| colombia_asia_hospital_bangalore    |
| fortis_chennai                      |
| hospital                            |
| jaslok_hospital_mumbai              |
| kolkata_medical_center_and_hospital |
| lisie_hospital_kochi                |
| lourdes_hospital_kochi              |
| mvn_group_of_hospitals_mumbai       |
| passwords                           |
| patient_list                        |
| sri_ramachandra_chennai             |
| vijaya_hospitals_chennai            |
+-------------------------------------+
14 rows in set (0.00 sec)

mysql> select * from colombia_asia_hospital_bangalore;
+------------+-------+-------+-------+-------+
| dov        | slot1 | slot2 | slot3 | slot4 |
+------------+-------+-------+-------+-------+
| 2022-03-13 |     5 |     5 |     5 |     5 |
| 2022-03-14 |     5 |     5 |     5 |     5 |
| 2022-03-15 |     5 |     5 |     5 |     5 |
| 2022-03-16 |     5 |     4 |     5 |     5 |
| 2022-03-17 |     5 |     5 |     5 |     5 |
| 2022-03-18 |     5 |     5 |     5 |     5 |
| 2022-03-19 |     5 |     5 |     5 |     5 |
+------------+-------+-------+-------+-------+
7 rows in set (0.00 sec)

mysql>
```

CONCLUSION

The process of developing this project has been a challenging yet interesting task. This topic had as steep learning curve which required our dedication and hard work to execute and achieve within the stipulated time. The application has turned out exactly as the initial plan with some added extra features.

This code of this project contains about 600 lines. Our aim was to make this project really simple so that even common people can use it easily.

We made login modules so that the safety of the users wont be compromised. Users can easily check the patient list of their account by logging in using their account.

The Python language is one of the most accessible programming languages available because it has simplified syntax and is not complicated, which gives more emphasis on natural language. Due to its ease of learning and usage, python codes can be easily written and executed much faster than other programming languages: Python language is efficient, reliable, and much faster than most modern languages.

One more best thing about the versatility of python language is that it can be used in many varieties of environments such as mobile applications, desktop applications, web development, hardware programming, and many more. The versatility of python makes it more attractive to use due to its high number of applications.

Now python language is being treated as the core programming language in schools and colleges due to its countless uses in Artificial Intelligence, Deep Learning, Data Science, etc. It has now become a fundamental part of the development world that schools and colleges cannot afford not to teach python language.