A Project Report On

"DISPENSARY MANAGEMENT"

Submitted By:

M. Pranav

School Roll No: 3

Class: XII C

CBSE Roll No:

Under the Guidance of

Mr. Anoop V S

PGT (Computer Science)

Department of Computer Science

SAINIK SCHOOL KALIKIRI

Department of Computer Science

SAINIK SCHOOL KALIKIRI

This is to certify that **Cdt. M PRANAV,** Roll No. 3 of Class XII has prepared the report on the Project entitled **“DISPENSARY MANAGEMENT”**. The report is the result of his efforts & endeavors. The report is found worthy of acceptance as final project report for the subject Computer Science of Class XII.

Signature	Signature
(Internal Examiner)	(External Examiner)

DECLARATION

I hereby declare that the project work entitled "**DISPENSARY MANAGEMENT**", submitted to Department of **Computer Science**, SAINIK SCHOOL KALIKIRI is **prepared by me**. All the **coding** is the result of my **personal efforts**.

Cdt. M. Pranav
Roll No: 3
Class: XII C
SAINIK SCHOOL KALIKIRI

ACKNOWLEDGEMENT

I would like to express a deep sense of thanks & gratitude to my **project guide Mr. Anoop V S** Sir for guiding me immensely through the course of the project. He always evinced keen interest in my work. His constructive advice & **constant motivation** have been responsible for the **successful** completion of this project.

My sincere thanks go to **Lt Col Susheel Kumar Mahapatro SM ,** our **Offg Principal** sir, for his co-ordination in extending every **possible support** for the completion of this project.

I also thanks to my **parents** for their **motivation & support**. I must thanks to my **classmates** for their timely help & support for **compilation** of this **project**.

Last but not the least, I would like to thank all those who had helped directly or indirectly towards the completion of this project.

Cdt. M. Pranav
Roll No: 3
Class: XII C
SAINIK SCHOOL KALIKIRI

CONTENTS

1. WORKING DESCRIPTION

LOGGED IN AS ADMIN

1. Manage Medicines

a. See the medicine List : will display the entire medicines table
b. Add a medicine : add a new medicine to the table
c. See expired Medicines : view the list of expired medicines
d. Update a medicine : update the data of the medicine
e. Delete an expired medicine : delete an expired medicine

2. Manage Admission / Discharge

a. Admit a cadet : admitting a cadet to the dispensary
b. Discharge a cadet : discharging a cadet from the dispensary
c. Extend discharge : ability to extend discharge period of cadet
d. See admission logs : will show all the admission logs
e. See discharge logs : will show all the discharge logs
f. See active admissions : will show all the active admissions

3. Issue Medicine

a. Issue Medicine : ability to issue medicine to the cadet
b. See list of cadets under medication : will show a table of cadet with their given medicines
c. See the list of all issued medicines : will show list of all medicines issues

4. See Cadets Data

a. List of all of the cadet's login : will show a cadet's login logs
b. List of a cadet login : will show a specific cadet logs
c. See cadet medications : ability to cadet history of medications
d. See cadet admissions/ discharges : ability to see cadet admissoins & discharges

5. Change your password : ability to change the admin password

LOGGED IN AS CADET

1. **Check you Logs :** will show the cadet logs

2. **See the Fit House Championship Leaderboard :** will show the table of caclulated points for the fit house championship

3. **Edit your Basic Medical Data**
    a. Update your height **:** ability to update cadet physical height
    b. Update your weight **:** ability to update cadet physical weight
    c. Update your eye sight **:** ability to update cadet eye sight
    d. Check your body mass index **:** will show the cadet BMI

4. **Change your password :** ability to change cadet password

2. Code of the Project

Note : There are 12 Python files in this project. Each file has it’s own importance in the project.

src.py : This is file has all the utility functions to run the complete code

```
import mysql.connector as sql
from mysql.connector import Error
from time import sleep
import os
import datetime
from prettytable import PrettyTable
from math import fabs

house_list = ['Godavari', 'Krishna', 'Penna', 'Tungabhadra']


def Establish_Connection():
    """This function establishes connection to mysql database"""
    connection = sql.connect(
        host='localhost', user='root', password='student', database='medic')
    mycursor = connection.cursor()

    return connection, mycursor, Error


# Establishing connection to the database
conn, cursor, sqlerror = Establish_Connection()


def Cls():
    """This function clears the clears the screen in the command prompt"""
    os.system('cls')


def Exit():
    """This function prints the Thank You message when a user exits the program"""
    print("Thank you for using the program")
    sleep(1.5)
    exit()


def Main_Heading():
    """This function prints the main heading"""
    print("---------- SAINIK SCHOOL KALIKIRI ---------- ")
    print("----------- Dispensary Management ---------- \n")


def Get_Admin_Username_List():
    """This function returns the list of admin usernames"""

    cursor.execute("SELECT username FROM admin_user")
    username_data = cursor.fetchall()
    username_list = []

    for row in username_data:
        username_list.append(row[0])

    return username_list
```

```
def Get_Admin_User_Password(username):
    """This function returns the password of a given user"""
    cursor.execute(
        f"SELECT password FROM admin_user WHERE username = '{username}'")
    password = cursor.fetchone()[0]

    return password


def Get_Admin_Name(username):
    """This function returns the name of the admin user given the username"""
    cursor.execute(
        f"SELECT name FROM admin_user WHERE username = '{username}'")
    name = cursor.fetchone()[0]

    return name


def Get_Cadet_Name(roll_no):
    """This function returns the name of the cadet based on his roll number"""
    cursor.execute(f"SELECT name FROM cadet WHERE roll_no = {roll_no}")
    name = cursor.fetchone()[0]

    return name


def Get_Roll_No_List():
    """This function returns the list of roll no of the cadets"""
    roll_list = []
    cursor.execute("SELECT roll_no FROM cadet")
    data = cursor.fetchall()

    for row in data:
        roll_list += row

    return roll_list


def Get_Cadet_Password(roll_no):
    """This function returns the password for a given Roll Number"""
    cursor.execute(
        f"SELECT password FROM cadet_user WHERE roll_no = {roll_no}")
    password = cursor.fetchone()[0]

    return password


def Redirecting():
    """This function just prints redirecting on the screen"""
    print("\nRedirecting", end='')
    sleep(1)
    print(".", end='')
    sleep(1)
    print(".", end='')
    sleep(1)
    print(".")


def Input_Date():
    """This function inputs date and returns the date"""
    print("\nPlease do enter only integers for the date")
```

```
while True:
        try:
            day_input = int(input("Enter the Day: "))
            month_input = int(input("Enter the Month: "))
            year_input = int(input("Enter the Year: "))

            the_date = datetime.date(year_input, month_input, day_input)

            return the_date

        except ValueError:
            print("\nYou have entered an invalid characters. Please Try Again")
            continue


def Check_Roll_No(roll_no):
    """This function checks whether the entered roll number
    exists in the data and returns True if matched"""
    roll_list = Get_Roll_No_List()

    if roll_no in roll_list:
        return True

    else:
        return False


def Get_Probable_Medicine(med_name):
    """This function return the probable list and table of medicines"""
    cursor.execute(
        f"SELECT medicine_name FROM medicine WHERE medicine_name like '%{med_name}%'")
    data = cursor.fetchall()

    med_list = []

    med_table = PrettyTable()
    med_table.field_names = ['Medicine Name']

    for med in data:
        med_table.add_row([med[0]])
        med_list.append(med[0])

    if len(med_list) == 0:
        return [], []

    else:
        return med_list, med_table


def Check_Expiry(med_name):
    """This function checks whether the given medicine in expired or not"""
    try:
        cursor.execute(
            f"SELECT expiry FROM medicine WHERE medicine_name = '{med_name}'")
        expiry_date = cursor.fetchone()[0]

        if expiry_date > datetime.date.today():
            return True

        else:
            return False

    except Error:
        return 'Error'
```

```
def Check_Quantity(med_name):
    """This function checks the quantity available to issue medicine"""
    cursor.execute(
        f"SELECT quantity FROM medicine WHERE medicine_name = '{med_name}'")
    aval_quantity = cursor.fetchone()[0]

    if aval_quantity > 0:
        return True

    else:
        return False


def Change_Medication_Status():
    """This function checks the end_date and updates the medical status of the cadet"""
    try:
        cursor.execute(
            "SELECT roll_no, timestamp, end_date, status FROM issue_medicine WHERE status = 'Under
Medication'")
        data = cursor.fetchall()

        for roll_no, timestamp, end_date, status in data:
            if end_date < datetime.date.today():
                cursor.excecute(
                    f"UPDATE issue_medicine SET status = 'Healthy' WHERE roll_no = {roll_no} and
timestamp = '{timestamp}'")
                conn.commit()

            else:
                pass

    except sqlerror:
        print("An Error Occurred while parsing and modifying the Status of the Cadet.")


def Scan_For_Expiry():
    """This function scans the entire medicines to check if any medicine expired"""
    expiry_table = PrettyTable()
    expiry_table.field_names = ['Medicine Name',
                                'Usage / Indication', 'Quantity', 'Expiry']
    expiry_list = []
    try:
        cursor.execute(f"SELECT medicine_name FROM medicine")
        data = cursor.fetchall()

        for medicine in data:
            check_value = Check_Expiry(medicine[0])
            if not check_value:
                cursor.execute(
                    f"SELECT * FROM medicine WHERE medicine_name = '{medicine[0]}'")
                name, usage, qty, exp = cursor.fetchone()
                expiry_list.append(name)
                expiry_table.add_row([name, usage, qty, exp])
            else:
                pass
        if len(expiry_list) == 0:
            return False, False
        else:
            return expiry_table, expiry_list

    except sqlerror:
        print("\nAn Error Occurred while scanning for expiry")
```

```
def Get_BMI_Status(bmi):
    """This function gets the bmi status based on the bmi value"""
    if bmi < 18.5 or bmi == 18.5:
        return 'Underweight'

    elif 18.5 < bmi < 24.9:
        return 'Healthy'

    elif 25 < bmi < 29.9:
        return 'Overweight'

    elif bmi > 30:
        return 'Obese'

    else:
        return "Can't Be Calculated"


def Update_BMI(roll_no):
    """This function updates BMI and BMI Status of a cadet"""
    try:
        import mysql.connector
        new = mysql.connector.connect(
            host='localhost', user='root', password='student', database='medic')
        new_cursor = new.cursor()
        new_cursor.execute(
            f"SELECT height, weight FROM medical_data WHERE roll_no = {roll_no}")
        height, weight = new_cursor.fetchone()

        if height and weight:
            height = height/100
            bmi = (weight / height**2)
            bmi = round(bmi, 2)
            bmi_status = Get_BMI_Status(bmi)

            new_cursor.execute(
                f"UPDATE medical_data SET BMI = {bmi}, BMI_status = '{bmi_status}' WHERE roll_no =
                                                                     {roll_no}")
            new.commit()
            return True

        else:
            return False

    except sqlerror:
        return False


def Input_Timing():
    """This function lets user to enter timing"""
    while True:
        hour_input = input("Enter the hour (24 Hour Format): ")
        minute_input = input("Enter the minutes: ")

        try:
            hour_input = int(hour_input)
            minute_input = int(minute_input)

            if 0 <= hour_input <= 23 and 0 <= minute_input <= 59:
                time = datetime.time(hour=hour_input, minute=minute_input)
                return time
            else:
                print("You input exceeded the limit. Please Try Again")
                sleep(1)
                continue

        except ValueError:
            print(
                "You have entered an invalid value for hours of minute. Please Try Again")
            sleep(1)
            continue


def Check_If_Admitted(roll_no):
    """This function checks whether a cadet is admitted or not"""
    try:
        cursor.execute(
            f"SELECT * FROM admission WHERE roll_no = {roll_no} and status = 'Admitted'")
        data = cursor.fetchone()

        if data is not None:
            print("\nYou can't Admit the Cadet. The cadet is already Admitted")
            print(f"Admitted Cause: {data[1]}")
            print(f"Admitted on: {data[3]}")
            print(f"Discharge Date: {data[2]}")
            return True

        else:
            return False

    except sqlerror:
        print("An Error Occurred while parsing data from the database. Please Try Again")
        sleep(1.5)


def Get_Latest_Timestamp(roll_no):
    """This function gets the latest admission timestamp of a cadet"""
    try:
        cursor.execute(
            f"SELECT timestamp FROM admission WHERE roll_no = {roll_no} and status = 'Admitted' ORDER
BY timestamp DESC")
        timestamp = cursor.fetchall()[0][0]

        return timestamp

    except sqlerror:
        print("\nAn error occurred while getting admission data. Please Try Again")
        sleep(1.5)
        return False

def Calculate_Eye_Sight_Points():
    """This functions calculates the points for eye sight"""
    for house in house_list:
        cursor.execute(f"""SELECT medical_data.eye_l,medical_data.eye_r FROM medical_data,cadet
                        WHERE cadet.house = '{house}' and cadet.roll_no = medical_data.roll_no""")
        data = cursor.fetchall()
        house_total = 0
        for eye_l, eye_r in data:
            if eye_l is None and eye_r is None:
                continue

            else:
                print("else is getting executed")
                eye_l = fabs(eye_l)
                eye_r = fabs(eye_r)
                eye_total = eye_l + eye_r
                house_total += eye_total

        else:
            cursor.execute(
                f"UPDATE fit_house SET eye_sight = {house_total} WHERE house = '{house}'")
            conn.commit()


def Calculate_BMI_Points():
    """This function calculates points based on BMI"""
    for house in house_list:
        cursor.execute(f"""SELECT medical_data.BMI FROM medical_data,cadet
                        WHERE cadet.house = '{house}' and cadet.roll_no = medical_data.roll_no""")
        data = cursor.fetchall()
        house_bmi = 0
        for row in data:
            bmi = row[0]

            if bmi is None:
                continue

            else:
                house_bmi += bmi

        else:
            cursor.execute(
                f"UPDATE fit_house SET BMI = {house_bmi} WHERE house = '{house}'")
            conn.commit()


def Add_Admission_Points(roll_no):
    """This function adds points the the fit house table if a cadet is admitted"""
    cursor.execute(f"SELECT house FROM cadet WHERE roll_no = {roll_no}")
    house = cursor.fetchone()[0]

    cursor.execute(
        f"UPDATE fit_house SET admission = admission + 5 WHERE house = '{house}'")
    conn.commit()

def Calculate_Total_Points():
    """This function calculates the total points for the house"""
    for house in house_list:
        cursor.execute(
            f"UPDATE fit_house SET total_points = bmi + eye_sight + admission WHERE house =
'{house}'")
        conn.commit()
```

main.py : This file must be run to start the program

```
import src
import admin_main as adm
import cadet_main as cdt
from time import sleep
from getpass import getpass

# Establishing connection to the database
conn, cursor, sqlerror = src.Establish_Connection()

def Validate_Admin():
    """This functions validates whether the user is admin"""
    print("\nYou chose to login as admin\n")
    username_input = input("\nEnter your username: ")  # Username Input
    username_list = src.Get_Admin_Username_List()

    if username_input in username_list:  # Validating Username
        password_input = getpass("Enter you password: ")  # Password Input
        user_password = src.Get_Admin_User_Password(username_input)

        if password_input == user_password:  # Validating Password
            print("\nYou have logged in as Admin Successfully\n")
            src.Redirecting()
            src.Cls()

            adm.Admin_Main()     # Redirecting to Admin if user in authorized

        else:  # Handling the invalid password
            print(
                "\nYou have entered an invalid password. Access Denied, Please Try Again\n")
            sleep(1.5)

    else:   # Handling Exception if username is not there
        print("\nYou have entered an invalid username. Please Try Again\n")
        sleep(1.5)


def Validate_Cadet():
    """This function validates whether the user is admin"""
    roll_no_input = input(
        "\nEnter you Roll Number: ")  # Taking Roll Number Input

    try:
        # Converting the Roll Number to Integer
        roll_no_input = int(roll_no_input)
        roll_no_list = src.Get_Roll_No_List()

        if roll_no_input in roll_no_list:  # Validating Roll Number
            # Password Input (Using getpass to avoid echoing of password)
            password = getpass("Enter you password: ")
            user_password = src.Get_Cadet_Password(roll_no_input)

            if password == user_password:  # Validating password
                print("\nYou have successfully logged in as Cadet")
                try:
                    # Adding a Cadet Log to the Database
                    cursor.execute(
                        f"INSERT INTO cadet_log VALUES ({roll_no_input}, CURRENT_TIMESTAMP())")
                    conn.commit()
                    src.Redirecting()
                    # Redirecting the user to the Cadet Menu
                    cdt.Cadet_Main(roll_no_input)
                except sqlerror:  # Handling the Database Exception
                    print(
                        "\nAn Error while sending data to the Database. Please Try Again")
                    sleep(1.5)

            else:  # Handling Password Exception
                print("\nYou have entered an Invalid Password, Please Try Again")
                sleep(1.5)
        else:  # Handling Roll Number Exception
            print("\nYou have entered an Invalid Roll No, Please Try Again")
            sleep(1.5)

    except ValueError:
        print("\nYou have not entered a number. Please Enter a Number")
        sleep(1.5)


def Main():
    """The main definition to start the program"""

    src.Cls()
    src.Main_Heading()

    while True:  # To make the options visible everytime
        # Printing the Main Menu
        print("---------------- Main Menu --------------- \n")
        print("Press (1) to log in as Admin")
        print("Press (2) to log in as Cadet")
        print("Press (3) to exit the program")

        # Dictionary to navigate to the required functions
        admin_dict = {'1':Validate_Admin,
                      '2': Validate_Cadet,
                      '3': src.Exit}

        main_input = input("Enter a valid input from the above options: ")

        # Validating main_input if true it will be navigated to the function
        if main_input in admin_dict:
            # Calling the function based on the dictionary
            admin_dict[main_input]()

        else:  # To avoid Error and display invalid message
            print("\nYou have entered an invalid input, please try again")
            sleep(1.5)
            continue


if __name__== '__main__':   # Running the program
    src.Calculate_BMI_Points()
    src.Calculate_Eye_Sight_Points()
    src.Calculate_Total_Points()
    change_issued_medicine_status()
    Main()
```

admin_main.py : This file executes the admin functionalities

```
import src
from time import sleep
from medicine import Medicine_Main
from admission_discharge import Admit_Discharge_Main
from issue_medicine import Issue_Medicine_Main
from cadet_log import Cadet_Log_Main
from admin_password import Admin_Password_Main


def Admin_Main():
    """This function prints the main menu for the admin user"""
    src.Cls()
    src.Main_Heading()

    while True:
        # Printing the Menu
        print("\n--------- Admin Menu -------- \n")
        print("Press (1) to Manage Medicines")
        print("Press (2) to Manage Admissions/Discharges")
        print("Press (3) to Issue Medicines")
        print("Press (4) to See the Cadets' Data")
        print("Press (5) to Change your Password")
        print("Press (6) to Go to the Main Menu")
        print("Press (7) to Exit the Program\n")

        # Admin Menu Dictionary to Navigate to specific Modules
        admin_menu_dict = {'1':Medicine_Main,
                           '2': Admit_Discharge_Main,
                           '3': Issue_Medicine_Main,
                           '4': Cadet_Log_Main,
                           '5': Admin_Password_Main,
                           '7': src.Exit}

        # Taking the input from the User
        admin_menu_input = input("Enter your input from the above options: ")

        if admin_menu_input in admin_menu_dict:
            # Calling the function based on the dictionary
            admin_menu_dict[admin_menu_input]()

        elif admin_menu_input == '6':
            print("\nYou chose to go to the Main Menu")
            sleep(1)
            break

        else:
            print("\nYou have entered an Invalid Input. Please Try Again")
            sleep(1.5)
            continue
```

cadet_main.py : This file executes the cadet functionalities

```
import src
from time import sleep
from cadet_password import Change_Cadet_Password_Main
from prettytable import PrettyTable
from basic_medical_data import Basic_Medical_Data_Main

# Establishing connection to the database
conn, cursor, sqlerror = src.Establish_Connection()



def Leaderboard(roll_no):
    """This function prints the leaderboard"""
    try:
        cursor.execute(
            f"SELECT house,total_points  FROM fit_house ORDER BY total_points desc")
        table = PrettyTable()
        table.field_names = ['Rank', 'House Name', 'Points']

        for i in range(1, 5):
            house, points = cursor.fetchone()
            if points is None:
                points = 0
            table.add_row([i, house, points])

        print(table)
        print("\n")
        sleep(1.5)

    except sqlerror:
        pass



def Check_Logs(roll_no):
    try:
        cursor.execute(f"SELECT timestamp FROM cadet_log WHERE roll_no = {roll_no} ORDER BY timestamp
                                                                                         desc")
        data = cursor.fetchall()
        entries = 0
        log_table = PrettyTable()
        log_table.field_names = ['Roll No', 'Time Stamp']

        for a in data:
            log_table.add_row([a])
            entries += 1

        print(log_table)
        print(f"\nYou have logged in {entries} Times")
        sleep(1.5)

    except sqlerror:
        print("\nAn Error Occurred while reading the data. Please Try Again")
        sleep(1.5)
```

```
def Cadet_Main(roll_no):
    """This function prints the Cadet Menu"""
    src.Cls()
    src.Main_Heading()

    while True:
        print("\n-------- Cadet Menu ------- \n")
        print("Press (1) to check your Logs")
        print("Press (2) to See Fit House Championship Leaderboard")
        print("Press (3) to Edit you Basic Medical Data")
        print("Press (4) to Change Your Password")
        print("Press (5) to go to the Main Menu")
        print("Press (6) to exit the Program\n")

        cadet_main_dict = {'1': Check_Logs,
                           '2': Leaderboard,
                           '3': Basic_Medical_Data_Main,
                           '4': Change_Cadet_Password_Main}

        cadet_main_input = input("Enter your input from the above options: ")

        if cadet_main_input in cadet_main_dict:
            # Calling the function based on the dictionary
            cadet_main_dict[cadet_main_input](roll_no)

        elif cadet_main_input == '5':
            print("\nYou chose to go the Main Menu")
            sleep(1.5)
            src.Cls()
            break

        elif cadet_main_input == '6':
            src.Exit()

        else:
            print("\nYou have entered an Invalid Input. Please Try Again")
```

medicine.py : This file executes functionalities related to medicines

```
import src
from time import sleep
from prettytable import PrettyTable
from update_medicine import Update_Medicine

conn, cursor, sqlerror = src.Establish_Connection() # Establishing connection to the database

def Medicine_List():
    """This function prints the list of medicine table"""
    cursor.execute(f"SELECT * FROM medicine") # Getting data from MySql
    medicine_data = cursor.fetchall()
    medicine_table = PrettyTable() # Creating Table named medicine_table
    medicine_table.field_names = ['Medicine Name','Usage / Indication','Quantity','Expiry Date']

    for a,b,c,d in medicine_data: # Adding Data to the medicine_table
        medicine_table.add_row([a,b,c,d])
    print("\n")
    print(medicine_table,'\n')
    sleep(2)
```

```
def Add_Medicine_Confirmation():
    """This function handles the Prerequisites before adding the new medicine to the database"""
    print("\n-- Add Medicine --\n")
    med_name_input = input("Enter the Medicine Name to Add: ")
    probable_med_list, probable_med_table = src.Get_Probable_Medicine(med_name_input)

    if med_name_input in probable_med_list:
        print("\nThe medicine you want to enter already exists. Try Updating Them")
        sleep(1.5)

    elif probable_med_list:
        print("\nProbable Medicine Table\n")
        print(probable_med_table)
        print("If you medicine is not in the probable list you can add them")
        confirm_add = input("Are you sure you want to add/update new medicine (Y / N): ")

        if confirm_add in ['y','Y']:
            Add_Medicine()

        else:
            print("\nYou chose not to add a new medicine")
            sleep(1.5)

    else:
        Add_Medicine()



def Add_Medicine():
    """This function gets required inputs and adds the new medicine to the database"""
    med_name = input("Please enter the medicine name again: ")
    usage = input("Enter the Usage / Indication of the Medicine: ")
    quantity = input("Enter the quantity of the Medicine: ")
    print("\nPlease Enter the Medicine Expiry Date Carefully")
    med_date =  src.Input_Date()

    try:
        cursor.execute(f"INSERT INTO medicine VALUES ('{med_name}','{usage}',
{quantity},'{med_date}')")
        conn.commit()

        print("\nYou have successfully added a new medicine to the Database.")
        sleep(1.5)

    except sqlerror:
        print("\nAn Error occurred while sending data to the medicine. Please Try Again")
        sleep(1.5)



def Update_Medicine_Confirmation():
    """This function manages the prerequisites before updating the medicine
    and redirects it the update_medicine module"""
    print("\n-- Update Medicine --")
    med_name = input("Enter the Medicine Name: ")
    probable_med_list, probable_med_table = src.Get_Probable_Medicine(med_name)

    if med_name in probable_med_list:
        Update_Medicine(med_name)
    elif probable_med_list:
        print("\nProbable Medicine Table\n")
        print(probable_med_table)
        print("If the medicine is not in the probable list you can update them")
        confirm_add = input("Are you sure you want to add new medicine (Y / N): ")

        if confirm_add in ['Y','y']:
            print("\nPlease Enter the Medicine Name as in the Table")
            second_med_name = input("Please Enter the medicine as in Table: ")

            if second_med_name in med_name:
                Update_Medicine(second_med_name)

            else:
                print("\nYou have entered an Incorrect Medicine Name. Try Again")

        else:
            print("\nYou chose not to update the medicine.")
            sleep(1.5)

    else:
        print("\nWe did not find any probable medicine for your input. Please Try Again")
        sleep(1.5)



def See_Expiry():
    print("\n-- See Expiry --\n")
    medicine_table, expiry_list = src.Scan_For_Expiry()

    if medicine_table:
        print(medicine_table)
        sleep(1.5)

    else:
        print("\nNo Medicines have expired Till Date.")
        sleep(1.5)

def Delete_Expired_Medicine():
    print("-- Delete Expired Medicine")
    medicine_table , expiry_list =src.Scan_For_Expiry()

    if medicine_table:
        print(medicine_table)

        med_input = input("Enter the Medicine you want to delete: ")

        if med_input in expiry_list:
            cursor.execute(f"DELETE FROM medicine WHERE medicine_name = '{med_input}'")
            conn.commit()

            print(f"You have successfully deleted {med_input} from the database.")
            sleep(1.5)

        else:
            print("You have entered an invalid medicine name to delete. Please Try Again.")
            sleep(1.5)
    else:
        print("There are no expired medicine. You don't need to delete any of them")
        sleep(1.5)



def Medicine_Main():
    """This function runs the Medicine Management Menu"""
    src.Cls()

    while True:
        print("\n-------- Medicine Management Menu-------- \n")
        print("Press (1) to see the Medicine List")
        print("Press (2) to Add a Medicine")
        print("Press (3) to See Expired Medicines")
        print("Press (4) to Update a Medicine")
        print("Press (5) to Delete a Expired Medicine")
        print("Press (6) to go to Admin Menu")
        print("Press (7) to Exit the Program\n")

        medicine_dict = {'1' :Medicine_List,
                         '2' : Add_Medicine_Confirmation,
                         '3' : See_Expiry,
                         '4' : Update_Medicine_Confirmation,
                         '5' : Delete_Expired_Medicine,
                         '7' : src.Exit}

        medicine_input = input("Enter a valid input from the above options: ")

        if medicine_input in medicine_dict:
            medicine_dict[medicine_input]()   # Calling the function based on the dictionary

        elif medicine_input == '6': # To break the loop for Admin Menu
            print("\nYou chose to go to the Admin Menu")
            sleep(1.5)
            break

        else:   # Handling Invalid input exception
            print("You have entered an invalid input, Please Try Again")
            sleep(1.5)
            continue
```

update_medicine.py : This file executes functionalities related to medicines

```
import src
from time import sleep

# Establishing connection to the database
conn, cursor, sqlerror = src.Establish_Connection()



def Update_Medicine_Name(med_name):
    print("\n-- Update Medicine Name --\n")
    new_med_name = input("Enter the New Medicine Name: ")

    try:
        cursor.execute(
            f"UPDATE medicine SET medicine_name = '{new_med_name}' WHERE medicine_name =
'{med_name}'")
        conn.commit()

        print("\nYou have successfully updated the name of the medicine.")
        sleep(1.5)

    except sqlerror:
        print(
            "\nAn Error Occurred while sending data. Please check the Medicine Name Again.")
        sleep(1.5)



def Update_Usage(med_name):
    print("\n-- Update Medicine Usage / Indication --\n")

    new_usage = input("Enter the New Usage / Indication: ")

    try:
        cursor.execute(
            f"UPDATE medicine SET usage = '{new_usage}' WHERE medicine_name = '{med_name}'")
        conn.commit()

        print("\nYou have successfully updated the Usage / Indication of the Medicine")
        sleep(1.5)

    except sqlerror:
        print("An Error occurred while sending the data to the database. Please Try Again")
        sleep(1.5)



def Update_Quantity(med_name):
    print("\n-- Update Quantity --\n")
    quantity = input("Enter the Updated quantity: ")

    try:
        quantity = int(quantity)

        try:
            cursor.execute(
                f"UPDATE medicine SET quantity = {quantity} WHERE medicine_name = '{med_name}'")
            conn.commit()

            print("\nYou have successfully updated the quantity")
            sleep(1.5)

        except sqlerror:
            print(
                "\nAn Error occurred while sending data to the database. Please Try Again")
            sleep(1.5)

    except ValueError:
        print("\nYou have entered an invalid value for quantity. Please Try Again")
        sleep(1.5)
```

```
def Update_Expiry(med_name):
    print("\n-- Updated Expiry Date --\n")
    print("Enter the new Expiry Date")
    new_date = src.Input_Date()
    try:
        cursor.execute(
            f"UPDATE medicine SET expiry = '{new_date}' WHERE medicine_name = '{med_name}'")
        conn.commit()

        print("\nYou have successfully updated the expiry date")
        sleep(1.5)

    except sqlerror:
        print("\nAn Error occurred while sending data to the database. Please Try Again")
        sleep(1.5)



def Update_Medicine(med_name):

    while True:
        print("\n-- Update Medicine --\n")
        print("Press (1) to Update the Medicine Name")
        print("Press (2) to Update the Usage / Indication")
        print("Press (3) to Update the Quantity")
        print("Press (4) to Update the Expiry Date")
        print("Press (5) to go back to Manage Medicine Menu")
        print("Press (6) to Exit the Program\n")

        update_dict = {'1': Update_Medicine_Name,
                       '2': Update_Usage,
                       '3': Update_Quantity,
                       '4': Update_Expiry}

        update_input = input("Enter a valid input from the Above Options: ")

        if update_input in update_dict:
            update_dict[update_input](med_name)

        elif update_input == '5':
            print("\nYou chose to go to the Manage Medicine Menu")
            sleep(1.5)
            break

        elif update_input == '6':
            src.Exit()
        else:
            print("\nYou have entered an invalid option. Please Try Again")
            sleep(1.5)
```

issue_medicine.py : This file executes functionalities related to medicines

```
import src
from time import sleep
from prettytable import PrettyTable


# Establishing connection to the database
conn, cursor, sqlerror = src.Establish_Connection()



def Issue_Medicine_Confirmation():
    print("\n --Issue Medicine --")
    i = 1
    roll_no = input("Enter the Roll Number: ")
    first_value = None
    second_value =None
    while True:


        try:
            roll_no = int(roll_no)
            med_name = input("Enter the Medicine Name: ")
            probable_medicine_list, probable_table = src.Get_Probable_Medicine(
                med_name)
            if med_name in probable_medicine_list:
                confirmation = input(
                    "\nAre you sure you want to issue the medicine (Y / N): ")
                if confirmation in ['Y', 'y']:
                    first_value = Expiry_Check(roll_no, med_name)


                else:
                    print("\nYou chose not to issue the medicine")
                    sleep(1.5)


            elif probable_medicine_list:
                print("\nProbable Medicine Table")
                print(probable_table)
                print(
                    "If the medicine is in the Probable list.Please Enter the Medicine Name as in the
                                                                                            Table")


                new_med_name = input("Please Enter the medicine as in Table: ")
                if new_med_name in probable_medicine_list:
                    confirm_input = input(
                        "Are you sure you want to issue this medicine (Y / N): ")


                    if confirm_input in ['Y', 'y']:
                        second_value = Expiry_Check(roll_no, new_med_name)


                    else:
                        print("\nYou chose not to issue the medicine")
                        sleep(1.5)
                else:
                    print(
                        "\nYou have entered the medicine name wrong for two times.Try Again\n")
                    sleep(1.5)
                    break


        except ValueError:
            print("\nYou have entered an invalid value for Roll No.Please Try Again")
            sleep(1.5)

        if first_value or second_value:
            next_medicine = input(
                f"You have issued {i} Medicine(s) to Roll No {roll_no}.Do you want to issue more(Y /
                                                                                              N): ")
            if next_medicine in ['Y', 'y']:
                i += 1
                continue

            else:
                print("\nYou have closed the issue medicine.\n")
                sleep(1.5)
                break
        else:
            print("\nYou have not issued any medicine yet! Try Again\n")
            sleep(1.5)
            break



def Expiry_Check(roll_no, med_name):

    check = src.Check_Expiry(med_name)

    if check:
        return Issue_Medicine(roll_no, med_name)

    else:
        print("\nThe medicine has expired. You cannot Issue the Medicine")
        sleep(1.5)
        return False



def Issue_Medicine(roll_no, med_name):
    qty = input(f"Enter the Quantity you want to Issue for '{med_name}': ")

    try:
        qty = int(qty)
        cause = input(f"Enter the Cause for Issuing Medicine '{med_name}': ")
        print("Enter the date by when the medicine should be consumed:")
        completion = src.Input_Date()

        try:

            cursor.execute(
                f"INSERT INTO issue_medicine VALUES ({roll_no},'{cause}','{med_name}',
                                {qty},CURRENT_TIMESTAMP(),'{completion}', 'Under Medication')")
            conn.commit()
            cursor.execute(
                f"UPDATE medicine SET quantity = quantity - {qty} WHERE medicine_name = '{med_name}'")
            conn.commit()

            print(
                f"\nYou have successfully issued '{med_name}' to Roll No {roll_no}")
            sleep(1.5)
            return True

        except sqlerror:
            print(
                "\nAn Error occurred while sending data to the database. Please Try Again")
            sleep(1.5)
            return False

    except ValueError:
        print("\nYou have entered an invalid value for Quantity. Please Try Again")
        sleep(1.5)
        return False



def Under_Medication():
    print("-- Under Medication List --")
    try:
        cursor.execute("select issue_medicine.roll_no, cadet.name, cadet.class, issue_medicine.cause, 
medicine, quantity, timestamp, end_date, status from issue_medicine natural join cadet where 
issue_medicine.status = 'Under Medication'")
        data = cursor.fetchall()
        med_table = PrettyTable()
        med_table.field_names = ['Roll No', 'Name', 'Class', 'Cause',
                                 'Medicine', 'Qty', 'Timestamp', 'End Date', 'Status']
        for row in data:
            med_table.add_row(row)

        print(med_table)
        print('\n')

    except sqlerror:
        print("An Error occurred while fetching the data. Please Try Again")
        return None



def See_Issued_Medicine():
    print("\n-- Issued Medicine Table\n")

    try:
        cursor.execute("""SELECT 
issue_medicine.roll_no,name,class,cause,medicine,quantity,timestamp,end_date,status
        FROM issue_medicine, cadet WHERE cadet.roll_no = issue_medicine.roll_no ORDER BY timestamp 
desc""")
        data = cursor.fetchall()
        table = PrettyTable()
        table.field_names = ['Roll No', 'Name', 'Class', 'Cause',
                             'Medicine', 'Qty', 'TimeStamp', 'End Date', 'Status']

        for roll_no, name, clas, cause, medicine, quantity, timestamp, end_date, status in data:
            table.add_row([roll_no, name, clas, cause, medicine,
                           quantity, timestamp, end_date, status])

        print(table)
        sleep(1.5)
        print('\n')

    except sqlerror:
        print("\nAn Error occurred while parsing the Data. Please Try Again.")
        sleep(1.5)
```

```
def change_issued_medicine_status():
    """This function changes the cadet's medication status as per End Date"""
    cursor.execute(
        "select * from issue_medicine where end_date < current_date() and status = 'Under
Medication'")
    data = cursor.fetchall()

    if data == []:
        return None

    updated_entries = 0
    for row in data:
        time_id = row[4]
        print(type(time_id))
        status = 'Healthy'

        cursor.execute(
            f"UPDATE issue_medicine set status = 'Healthy' where timestamp = '{time_id}'")
        conn.commit()
        updated_entries += 1

    print("Number of updated Entries =", updated_entries)



def Issue_Medicine_Main():
    src.Cls()

    while True:
        print("------- Issue Medicine Menu ------ \n")
        print("Press (1) to Issue Medicine")
        print("Press (2) to see list of Cadet's Under Medication")
        print("Press (3) to see the list of all Issued Medicines")
        print("Press (4) to go to Admin Menu")
        print("Press (5) to exit the Program\n")

        issue_dict = {'1': Issue_Medicine_Confirmation,
                      '2': Under_Medication,
                      '3': See_Issued_Medicine,
                      '5': src.Exit}

        issue_input = input("Enter a valid input from the available options: ")

        if issue_input in issue_dict:
            # Calling the function based on the dictionary
            issue_dict[issue_input]()

        elif issue_input == '4':
            print("\nYou chose to go the Admin Menu")
            sleep(1.5)
            break

        else:
            print("\nYou have entered an Invalid Input. Please Try Again\n")
            sleep(1.5)
            continue
```

admission_discharge.py : This file executes the functionalities related to admissions and discharges

```
from time import sleep
import src
from prettytable import PrettyTable

# Establishing connection to the database
conn, cursor, sqlerror = src.Establish_Connection()


def Admit_Cadet():
    """This function lets the admin user to admit a cadet"""
    print("\n--- Admit Cadet ---\n")
    roll_no_input = input("Enter the Roll No of the Cadet: ")
    try:
        roll_no_input = int(roll_no_input)
        roll_no_check = src.Check_Roll_No(roll_no_input)

        if roll_no_check:
            reason = input("Enter the reason for Admission: ")
            print("Enter the Date of the Discharge")
            discharge_date = src.Input_Date()
            confirmation = input("\nPlease Confirm the admission (Y / N): ")

            if confirmation in ['Y', 'y']:

                database_confirmation = src.Check_If_Admitted(roll_no_input)

                if not database_confirmation:

                    try:
                        cursor.execute(
                            f"INSERT INTO admission VALUES
       ({roll_no_input},'{reason}','{discharge_date}',CURRENT_TIMESTAMP(),'Admitted',Null)")
                        conn.commit()
                        src.Add_Admission_Points(roll_no_input)
                        cursor.execute(
                            f"SELECT name FROM cadet WHERE roll_no = {roll_no_input}")
                        cadet_name = cursor.fetchone()[0]
                        print(f"\nYou have successfully admitted {cadet_name}")
                        sleep(1.5)

                    except sqlerror:
                        print(
                            "\nAn Error occurred while sending data to database.Please Try Again")
                        sleep(1.5)

                else:
                    print("\nThe cadet can't be admitted")
                    sleep(1.5)

            else:
                print("\nYou have cancelled the admission.\n")
                sleep(1.5)

        else:
            print("\nThe Entered Roll Number does not Exists. Please Try Again\n")
            sleep(1.5)

    except ValueError:
        print(
            "\nYou have not entered a correct value for the Roll Number. Please Try Again")
        sleep(1.5)


def Discharge_Cadet():
    """This function lets admin user to discharge a cadet"""
    print("\n-- Discharge Cadet --\n")

    try:
        cursor.execute(f"""SELECT admission.roll_no, cadet.name, admission.cause,
admission.discharge_date
                        FROM admission, cadet WHERE cadet.roll_no = admission.roll_no and
admission.status = 'Admitted'""")
        data = cursor.fetchall()

        if data is None:
            print("No cadet is admitted. You can't discharge anyone.\n")
            sleep(1.5)

        else:
            table = PrettyTable()
            table.field_names = ['Roll No', 'Name', 'Cause', 'Discharge Date']
            roll_no_list = []
            for no, name, cause, date in data:
                table.add_row([no, name, cause, date])
                roll_no_list.append(no)

            print(table)

            roll_no = input(
                "Enter the Roll Number of the Cadet to Discharge: ")
            try:
                roll_no = int(roll_no)
                if roll_no in roll_no_list:

                    try:
                        cadet_name = src.Get_Cadet_Name(roll_no)
                        confirm = input(
                            f"Are you sure you want to discharge {cadet_name} (Y / N): ")

                        if confirm in ['y', 'Y']:
                            timestamp = src.Get_Latest_Timestamp(roll_no)

                            try:
                                cursor.execute(f"""UPDATE admission SET status = 'Discharged',
                                                discharge_timestamp = current_timestamp() WHERE
                                                    roll_no = {roll_no} and
                                                timestamp = '{timestamp}'""")
                                conn.commit()

                                print(
                                    f"\nYou have successfully discharged {cadet_name}")
                                sleep(1.5)

                            except sqlerror:
                                print(
                                    "\nAn Error occurred while updating the data. Please Try Again")
                                sleep(1.5)

                        else:
                            print(f"\nYou chose not to discharge {cadet_name}")
                            sleep(1.5)
                    except ValueError:
                        print(
                            "\nYou have entered an invalid value for Roll Number. Please Try Again")
                        sleep(1.5)

                else:
                    print(
                        "\nYou can't discharge the cadet as the cadet is not admitted.")
                    sleep(1.5)
            except ValueError:
                print("You have entered an invalid Roll No. Please Try Again")
                sleep(1.5)

    except sqlerror:
        print("\nAn Error Occurred while parsing the Admission Data. Please Try Again\n")
        sleep(1.5)


def Extend_Discharge_Confirmation():
    """This function takes the required input to extend the discharge date of the cadet"""
    print("\n-- Extend Discharge --\n")
    Active_Admissions()
    roll_no_input = input(
        "Enter the Roll Number of the cadet to extend Discharge: ")

    try:
        roll_no_input = int(roll_no_input)

        confirm = input("\nDo you want to extend the discharge date (Y / N): ")

        if confirm in ['y', 'Y']:
            Extend_Discharge(roll_no_input)

        else:
            print(f"You chose not to extend the discharge of the cadet")

    except ValueError:
        print("\nAn Error Occurred while evaluating roll number or connecting to the database. ")
        sleep(1.5)


def Extend_Discharge(roll_no):
    """This function extends the discharge date of a admitted cadet"""
    print("\nEnter the new data for discharge")
    new_date = src.Input_Date()
    timestamp = src.Get_Latest_Timestamp(roll_no)

    try:
        cursor.execute(f"""UPDATE admission SET discharge_date = '{new_date}' WHERE roll_no =
{roll_no} AND
                        status = 'Admitted' and timestamp = '{timestamp}'""")
        conn.commit()

        print("\nYou have successfully updated the discharge date")

    except sqlerror:
        print("\nAn Error Occurred while sending the data. Please Try Again")
        sleep(1.5)


def Admission_Logs():
    """This function prints the admission logs to the admin user"""
    print("\n-- Admission Logs --\n")
    try:
        cursor.execute(f"""SELECT admission.*,cadet.name FROM admission,cadet WHERE admission.roll_no
= cadet.roll_no
                        ORDER BY timestamp DESC""")
        data = cursor.fetchall()

        table = PrettyTable()
        table.field_names = ['Roll No', 'Name', 'Cause',
                             'Discharge On', 'Admission Time','Status']

        for roll, cause, discharge, timestamp, status, discharge_time, name in data:
            table.add_row([roll, name, cause, discharge, timestamp, status])

        if data is None:
            print("\nThere are no admission Logs to display")
            sleep(1.5)

        else:
            print(table)
            print("\n")

    except sqlerror:
        print(
            "\nAn Error Occurred while receiving data from the database. Please Try Again")
        sleep(1.5)


def Discharge_Logs():
    """This functions prints all the discharge logs to the admin user"""
    print("\n-- Discharge Logs --\n")
    try:
        cursor.execute(f"""SELECT admission.*, cadet.name FROM admission, cadet WHERE
admission.roll_no = cadet.roll_no
                        AND discharge_timestamp IS NOT NULL AND status = 'Discharged'
                        ORDER BY discharge_timestamp DESC""")
        data = cursor.fetchall()
        table = PrettyTable()
        table.field_names = ['Roll No', 'Name', 'Cause',
                             'Discharge Date', 'Admission Time', 'Status', 'Discharged On']

        for roll, cause, discharge, timestamp, status, discharge_time, name in data:
            table.add_row([roll, name, cause, discharge,
                           timestamp, status, discharge_time])

        if data is None:
            print("\nThere are discharges to show")
            sleep(1.5)

        else:
            print(f"\n{table}")
            sleep(1.5)

    except sqlerror:
        print(
            "\nAn error occurred while receiving data from the database. Please Try Again")
        sleep(1.5)


def Active_Admissions():
    """This function prints all the active admissions to the admin user"""
    print("\n-- Active Admissions --")
    try:
        cursor.execute(f"""SELECT admission.roll_no, cadet.name, admission.cause,
admission.discharge_date,
                        admission.timestamp FROM admission, cadet WHERE cadet.roll_no =
admission.roll_no and
                        status = 'Admitted'""")
        data = cursor.fetchall()

        table = PrettyTable()
        table.field_names = ['Roll No', 'Name', 'Cause',
                             'Discharge Date', 'Time of Admission']

        for no, name, cause, discharge, time in data:
            table.add_row([no, name, cause, discharge, time])

        if data is None:
            print("\nThere are No Active Admissions")
            sleep(1.5)

        else:
            print(table)
            print('\n')

    except sqlerror:
        print("\nAn Error occurred while parsing the data. Please Try Again.")


def Admit_Discharge_Main():
    """This function prints the menu of the discharge to the admin user"""
    src.Cls()

    while True:
        print("\n---------- Admission/Discharge Menu ---------\n")
        print("Press (1) to Admit a Cadet")
        print("Press (2) to Discharge Cadet")
        print("Press (3) to Extend Discharge")
        print("Press (4) to see Admission Logs")
        print("Press (5) to see Discharge Logs")
        print("Press (6) to see Active Admissions")
        print("Press (7) to go to Admin Menu")
        print("Press (8) to Exit the Program\n")

        admit_dict = {'1': Admit_Cadet,
                      '2': Discharge_Cadet,
                      '3': Extend_Discharge_Confirmation,
                      '4': Admission_Logs,
                      '5': Discharge_Logs,
                      '6': Active_Admissions,
                      '8': src.Exit}

        admit_input = input("Enter a Valid input from the above options: ")

        if admit_input in admit_dict:
            # Calling the function based on the dictionary
            admit_dict[admit_input]()

        elif admit_input == '7':
            print("\nYou chose to go the Admin Menu")
            sleep(1.5)
            break

        else:
            print("\nYou have entered an Invalid Input. Please Try Again")
            sleep(1.5)
            continue
```

basic_medical_data.py : This file execute the cadet functionalities of managing cadet medical data

```
import src
from time import sleep

conn, cursor, sqlerror = src.Establish_Connection() # Establishing connection to the database

def Height_Update(roll_no):
    """This function shows and updates the height of the cadet"""
    try:
        cursor.execute(f"SELECT height FROM medical_data WHERE roll_no = {roll_no}")
        present_height = cursor.fetchone()[0]

        if present_height is None:
            update_input = 'Y'

        else:
            print(f"\nYour Height according to the database is : {present_height}")
            update_input = input(f"Do you want to change your height (Y / N): ")

        if update_input in ['y','Y']:
                print("\nYou chose to update your height")
                print("Example : 156.20")
                new_height = input("Enter you height in centimeters: ")

                try:
                    new_height = float(new_height)
                    new_height = round(new_height,2)
                    cursor.execute(f"UPDATE medical_data SET height = {new_height}  WHERE roll_no =
                                                                                    {roll_no}")
                    conn.commit()

                    value = src.Update_BMI(roll_no)

                    if value:
                        print("\nYour BMI is updated")
                        sleep(1.5)

                    else:
                        print("\nYour BMI is not updated check your Weight")
                        sleep(1.5)

                except ValueError:
                    print("\nYou have entered an Invalid value for the height. Please Try Again")
                    sleep(1.5)

        else:
            print("\nYou chose not to update your height")
            sleep(1.5)

    except sqlerror:
        pass
```

```
def Weight_Update(roll_no):
    """This function shows and updates the weight of the cadet"""
    try:
        cursor.execute(f"SELECT weight FROM medical_data WHERE roll_no = {roll_no}")
        present_weight = cursor.fetchone()[0]

        if present_weight is None:
            update_input ='Y'

        else:
            print(f"\nYour Weight according to the database is : {present_weight}")
            update_input = input(f"Do you want to change your weight (Y / N): ")

        if update_input in ['y', 'Y']:
            print("\nYou chose to update your weight")
            print("Example : 66.20")
            new_weight = input("Enter you weight in kilograms: ")

            try:
                new_weight = float(new_weight)
                new_weight = round(new_weight, 2)
                cursor.execute(f"UPDATE medical_data SET weight = {new_weight} WHERE roll_no =
                                                                                   {roll_no}")
                conn.commit()

                value = src.Update_BMI(roll_no)

                if value:
                    print("\nYour BMI is updated")
                    sleep(1.5)

                else:
                    print("\nYour BMI is not updated check your Height")
                    sleep(1.5)

            except ValueError:
                print("\nYou have entered an Invalid value for the weight. Please Try Again")
                sleep(1.5)

        else:
            print("\nYou chose not to update your weight")
            sleep(1.5)

    except sqlerror:
        pass


def Eye_Sight_Confirm(roll_no):
    """This function confirms whether a cadet is having eye sight or not"""

    try:
        cursor.execute(f"SELECT eye_l, eye_r FROM medical_data WHERE roll_no = {roll_no}")
        left, right = cursor.fetchone()

        if left and right is None:
            print("\nYou have not entered you Eye Sight till now.Please Update")
            sleep(1)
            Eye_Sight_Update(roll_no)

        else:
            print("Your Present Eye Sight")
            print(f"Left Eye : {left}")
            print(f"Right Eye: {right}\n")

            confirmation = input("Do you want to update your Eye Sight (Y / N): ")
            if confirmation in ['y','Y']:
                Eye_Sight_Update(roll_no)

            else:
                print("\nYou chose not to update your eye sight")
                sleep(1.5)

    except sqlerror:
        print("\nAn error occurred while parsing the data from the database. Please Try Again")
        sleep(1.5)

def Eye_Sight_Update(roll_no):
    """This function updates eye sight of the cadet"""
    confirm_input = input("Do you have Eye Sight (Y / N): ")

    if confirm_input in ['y','Y']:
        print("\nExample: -1.25, ")
        print("If you have perfect vision for an Eye. Please Enter Zero\n")
        r_eye = input("Enter your Left Eye Sight: ")
        l_eye = input("Enter your Right Eye Sight: ")

        try:
            cursor.execute(f"UPDATE medical_data SET eye_r = {r_eye}, eye_l = {l_eye} WHERE roll_no =
{roll_no}")
            conn.commit()

            print("\nYou have successfully updated your eye sight")
            sleep(1.5)

        except sqlerror:
            print("\nAn error occurred while sending the data to the database")
            sleep(1.5)

    else:
        print("\nYou don't have Eye Sight")
        sleep(1.5)



def BMI_Check(roll_no):
    """This function prints the cadet's BMI and other data"""
    print("\nYou chose to see your BMI")
    try:
        cursor.execute(f"""SELECT medical_data.roll_no, cadet.name, cadet.class, medical_data.height,
medical_data.weight,
                        medical_data.BMI, medical_data.BMI_status FROM medical_data, cadet
                        WHERE medical_data.roll_no = cadet.roll_no and cadet.roll_no = {roll_no}""")
        roll,name,clas, height, weight, bmi, bmi_status = cursor.fetchone()

        print(f"Roll Number: {roll_no}")
        print(f"Cadet Name: {name}")
        print(f"Class: {clas}")
        print(f"Cadet height: {height}")
        print(f"Cadet Weight: {weight}")
        print(f"Cadet BMI: {bmi}")
        print(f"Cadet BMI Status: {bmi_status}\n")
        sleep(2)

    except sqlerror:
        print("\nAn Error occurred while parsing the data. Please Try Again")
        sleep(1.5)

def Basic_Medical_Data_Main(roll_no):
    """This function prints the menu of the Updating of the Medical Data"""

    src.Cls()
    while True:
        print("\n-- Personal Medical Data --\n")
        print("Press (1) to update your height")
        print("Press (2) to update your weight")
        print("Press (3) to update your eye sight")
        print("Press (4) to check you Body Mass Index")
        print("Press (5) to go to Cadet Menu")
        print("Press (6) to exit the program\n")

        medical_dict = {'1' : Height_Update,
                        '2' : Weight_Update,
                        '3'  : Eye_Sight_Confirm,
                        '4'  : BMI_Check}

        medical_input = input("Enter your input from the available options: ")

        if medical_input in medical_dict:
            medical_dict[medical_input](roll_no)

        elif medical_input == '5':
            print("\nYou chose to go to Cadet Menu")
            sleep(1.5)
            break

        elif medical_input == '6':
            src.Exit()

        else:
            print("\nYou have entered an invalid input. Please Try Again")
            sleep(1.5)
            continue
```

cadet_log.py : This file handles the cadets logs

```
import src
from time import sleep
from prettytable import PrettyTable


def List_Of_Cadet_Login():
    print("-- List of Cadets' Logins --")
    conn, cursor, sqlerror = src.Establish_Connection()
    rollno = input("Enter the Roll Number to see Log Ins : ")
    try:
        rollno =int(rollno)
        try:

            table = PrettyTable()
            table.field_names = ['Roll No', 'Name', 'Class', 'Timestamp']
            cursor.execute(
                f"SELECT cadet_log.roll_no, name ,class, timestamp from cadet_log,cadet WHERE
cadet_log.roll_no = {rollno} and cadet.roll_no = cadet_log.roll_no ORDER BY timestamp desc")
            data = cursor.fetchall()

            for row in data:
                table.add_row(row)
            print(table)
            print("\n")
            sleep(1.5)

        except sqlerror:
            print("\nAn Error Occurred while parsing the data. Please Try Again")
            sleep(1.5)
    except ValueError:
        print("\n<{rollno}> is invalid. Please Try Again")
        sleep(1.5)
    conn.close()



def List_Of_Cadets_Logins():
    conn, cursor, sqlerror = src.Establish_Connection()
    query = "select cadet_log.roll_no, cadet.name, count(cadet_log.roll_no) as 'Logged in Times',
cadet.class, cadet.section, cadet.house from cadet_log natural join cadet group by cadet_log.roll_no"
    try:
        cursor.execute(query)
        data = cursor.fetchall()
        table = PrettyTable()
        table.field_names = ['Roll No', 'Name',
                             'No of Times logged in', 'Class', 'Section', 'House']
        for row in data:
            table.add_row(row)

        print(table)
        print("\n")
        sleep(1.5)

    except sqlerror:
        print("\nAn Error Occurred while parsing the data. Please Try Again")
        sleep(1.5)
    conn.close()
```

```
def Cadet_Medications():
    print("-- Cadet Medications --")
    rollno = input("Enter the roll no : ")
    conn, cursor, sqlerror = src.Establish_Connection()
    try:
        rollno = int(rollno)
        cursor.execute(f"SELECT * FROM cadet WHERE roll_no = {rollno}")
        cadet_data = cursor.fetchone()
        try:
            cursor.execute(
                f"select * from issue_medicine where roll_no = {rollno}")
            medicine_data = cursor.fetchall()
            table = PrettyTable()
            table.field_names = ['Roll No', 'Cause', 'medicine',
                                 'quantity', 'timestamp', 'end_date', 'status']
            if len(cadet_data) == 0:
                print('The entered roll no does not exists. Please Try Again')
                conn.close()
                return None
            print("-----------------------")
            print(f"Roll No : {cadet_data[0]}")
            print(f"Name  : {cadet_data[1]}")
            print(f"Class  : {cadet_data[2]}")
            print(f"Section : {cadet_data[3]}")
            print(f"House  : {cadet_data[4]}")
            print("-----------------------")
            if medicine_data == []:
                print("Nothing to show. No data is found for the cadet.")
                conn.close()
                return None
            for row in medicine_data:
                table.add_row(row)
            print(table)
            sleep(1.5)
        except sqlerror:
            print("An Error Occurred while parsing the Data. Please Try Again")
            sleep(1.5)


    except ValueError:
        print("You have entered an invalid roll no. Please Try Again")
        sleep(1.5)
    conn.close()



def Cadet_Admissions():
    conn, cursor, sqlerror = src.Establish_Connection()
    rollno = input("Enter the Roll Number to see the Admissions : ")
    try:
        rollno = int(rollno)
        cursor.execute(f"SELECT * FROM cadet WHERE roll_no = {rollno}")
        cadet_data = cursor.fetchone()
        if len(cadet_data) == 0:
            print(f"<{rollno}> does not exists. Please Try Again")
            conn.close()
            return None
        print(" ---------------------- ")
        print(f"Roll No : {cadet_data[0]}")
        print(f"Name  : {cadet_data[1]}")
        print(f"Class  : {cadet_data[2]}")
        print(f"Section : {cadet_data[3]}")
        print(f"House  : {cadet_data[4]}")
        print(" ---------------------- ")
        try:
            cursor.execute(f"select * from admission where roll_no = {rollno}")
            data = cursor.fetchall()
            table = PrettyTable()
            table.field_names = [
                'Roll No', 'Cause', 'Discharge Date', 'Timestamp', 'Status', 'Discharge Timing']
            for row in data:
                table.add_row(row)

            print(table)
            print('\n')
            sleep(1.5)

        except sqlerror:
            print("An Error Occurred while parsing the data. Please Try Again")
            sleep(1.5)

    except ValueError:
        print(f"<{rollno}> is an invalid input. Please Try Again")
        sleep(1.5)
    conn.close()



def Cadet_Log_Main():
    src.Cls()

    while True:

        print("-------- Cadet Log Menu ------ \n")
        print("Press (1) to see List of Cadets' Login")
        print("Press (2) to see List of Cadet Login")
        print("Press (3) to see Cadet Medications")
        print("Press (4) to see Cadet Admissions/Discharges")
        print("Press (5) to go to Admin Menu")
        print("Press (6) to Exit the Program\n")

        cadet_log_dict = {'1': List_Of_Cadets_Logins,
                          '2': List_Of_Cadet_Login,
                          '3': Cadet_Medications,
                          '4': Cadet_Admissions,
                          '6': src.Exit}

        cadet_log_input = input("Enter your input from the above options: ")

        if cadet_log_input in cadet_log_dict:
            # Calling the function based on the dictionary
            cadet_log_dict[cadet_log_input]()
        elif cadet_log_input == '5':  # Taking to Admin Menu
            print("\nYou chose to go the Admin Menu")
            sleep(1.5)
            break
        else:
            print("\nYou have entered an Invalid Input. Please Try Again")
            continue
```

admin_password.py : This file handles the passwords of the admin users

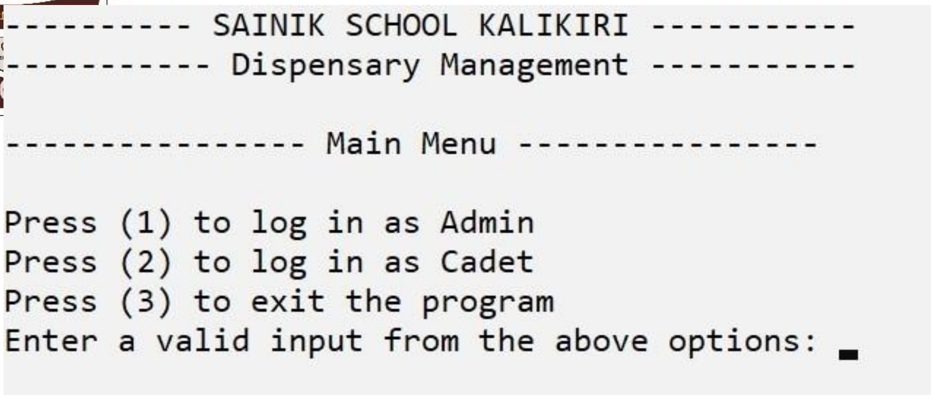
```
import src
from time import sleep
from getpass import getpass

conn, cursor, sqlerror = src.Establish_Connection() # Establishing connection to the database

def Change_Password(username):
    """This function helps the admin user to change his password"""
    new_pass_1 = getpass("Enter your new password: ")
    new_pass_2 = getpass("Please Enter you new password Again: ")

    if new_pass_1 == new_pass_2:
        try:
            cursor.execute(f"UPDATE admin_user SET password = '{new_pass_1}' WHERE username =
'{username}'")
            conn.commit()

        except sqlerror:
            print("\nAn Error occurred while sending data to the database. Please Try Again")
            sleep(1.5)

    else:
        print("\nThe Entered Passwords Do Not Match. Please Try Again.")
        sleep(1.5)

def Admin_Password_Main():
    """This function verifies if the user is authorized to change his password"""
    print("\n-- Change Password --\n")
    print("First Prove your Identity to change your password")

    username = input("Enter your Username: ")
    username_list = src.Get_Admin_Username_List()
    if username in username_list:
        password = getpass("Enter your Password: ")
        table_password = src.Get_Admin_User_Password(username)

        if password in table_password:
            print("\nYou are authorized to change you password\n")
            sleep(1.5)
            Change_Password(username)



        else:
            print("Incorrect Password was Entered. Please Try Again.")

    else:
        print("\nSorry. The Entered username does not exists. Please Try Again.")
        sleep(1.5)
```

cadet_password.py : This file handles the passwords of the cadet users

```
import src
from time import sleep
from getpass import getpass

conn, cursor, sqlerror = src.Establish_Connection() # Establishing connection to the database

def Change_Password(roll_no):
    """This function gets the new password and changes the password in the database"""
    pass_1 = getpass("Enter your new Password: ") # Getting new Password
    pass_2 = getpass("Enter your new Password Again: ")

    if pass_1 == pass_2: # checking if both the passwords match
        try:
            #Sending the data to the database 'medic'
            cursor.execute(f"UPDATE cadet_user SET password = '{pass_1}' WHERE roll_no = {roll_no}")
            conn.commit()

            print("\nYou have successfully changed your password") # Printing the success message
            sleep(1.5)

        except sqlerror: # Handling the MySql Exception
            print("\nAn Error Occurred while sending data to the database. Please Try Again")
            sleep(1.5)

    else: # Printing Error Message when passwords do not match
        print("\nThe entered passwords do not match. Please Try Again")
        sleep(1.5)



def Change_Cadet_Password_Main(roll_no):
    """This function authorizes user for changing password"""

    print("\n-- Change Password --\n") # Printing the heading
    print("First Prove your Identity to change your password\n")



    roll_list = src.Get_Roll_No_List()

    if roll_no in roll_list: # checking if the roll number exists in the database
        password = getpass("Enter your password: ")
        cadet_password = src.Get_Cadet_Password(roll_no)
        if password == cadet_password: # checking if the entered password matches with the database
            print("\nYou are authorized to change you password\n")
            sleep(1.5)
            Change_Password(roll_no)   #Authorizing user for changing password
        else: # Handling Invalid password Exception
            print("You have entered an invalid password. Please Try Again")
            sleep(1.5)
    else: # Handling Invalid Roll Number Exception
        print("The entered Roll Number does not exists. Please Try Again")
        sleep(1.5)
```

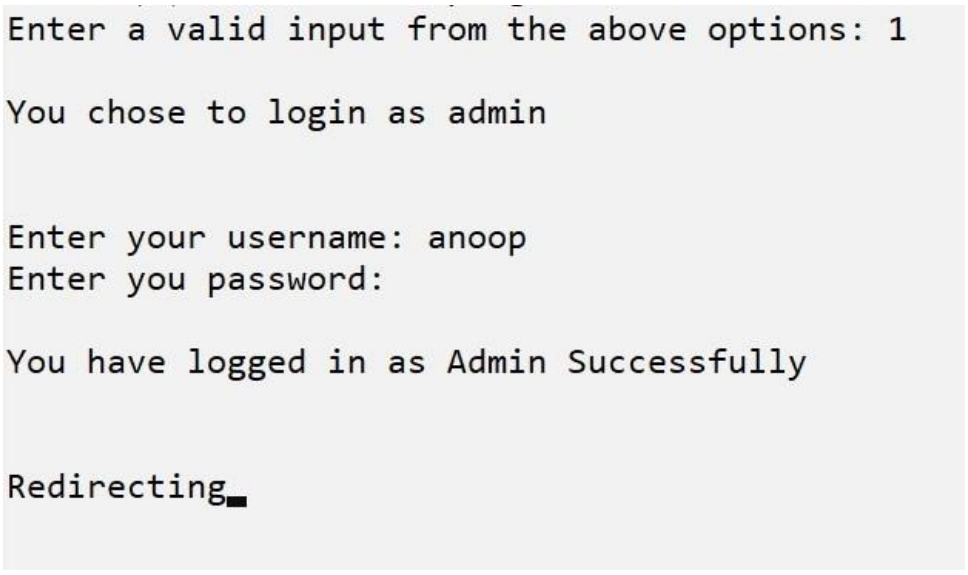
3. Output Screens

The main menu of the program



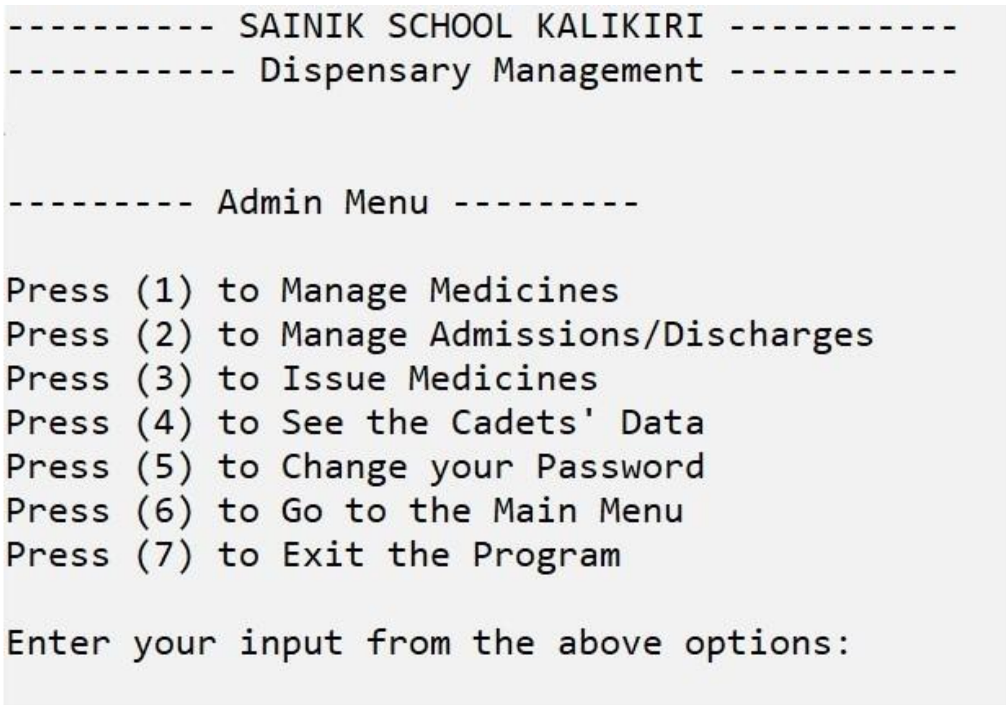
```
---------- SAINIK SCHOOL KALIKIRI -----------
----------- Dispensary Management -----------

---------------- Main Menu ----------------

Press (1) to log in as Admin
Press (2) to log in as Cadet
Press (3) to exit the program
Enter a valid input from the above options: _
```

Logging in as Admin



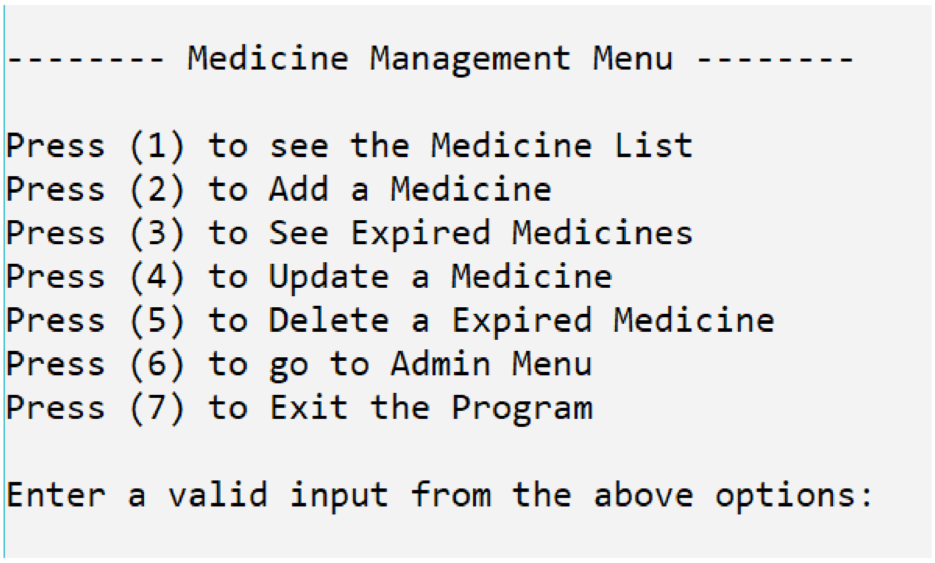
```
Enter a valid input from the above options: 1

You chose to login as admin


Enter your username: anoop
Enter you password:

You have logged in as Admin Successfully


Redirecting_
```

Admin Menu



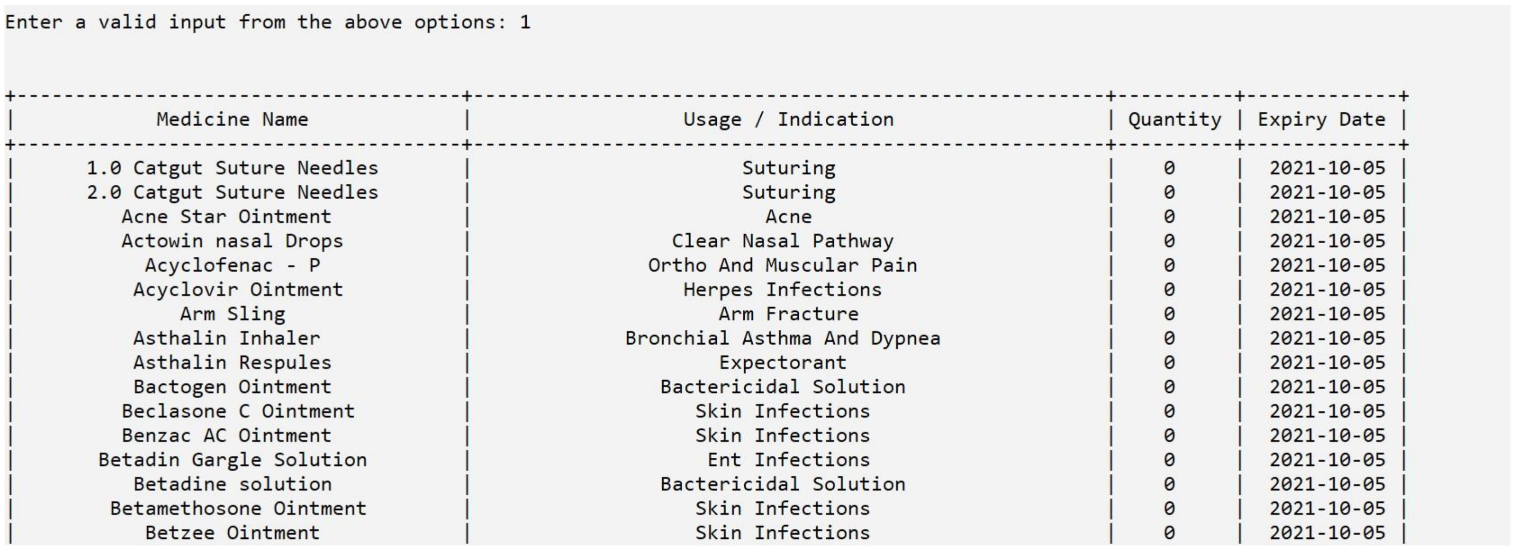
```
---------- SAINIK SCHOOL KALIKIRI -----------
----------- Dispensary Management -----------


--------- Admin Menu ---------

Press (1) to Manage Medicines
Press (2) to Manage Admissions/Discharges
Press (3) to Issue Medicines
Press (4) to See the Cadets' Data
Press (5) to Change your Password
Press (6) to Go to the Main Menu
Press (7) to Exit the Program

Enter your input from the above options:
```

Medicine Management Menu



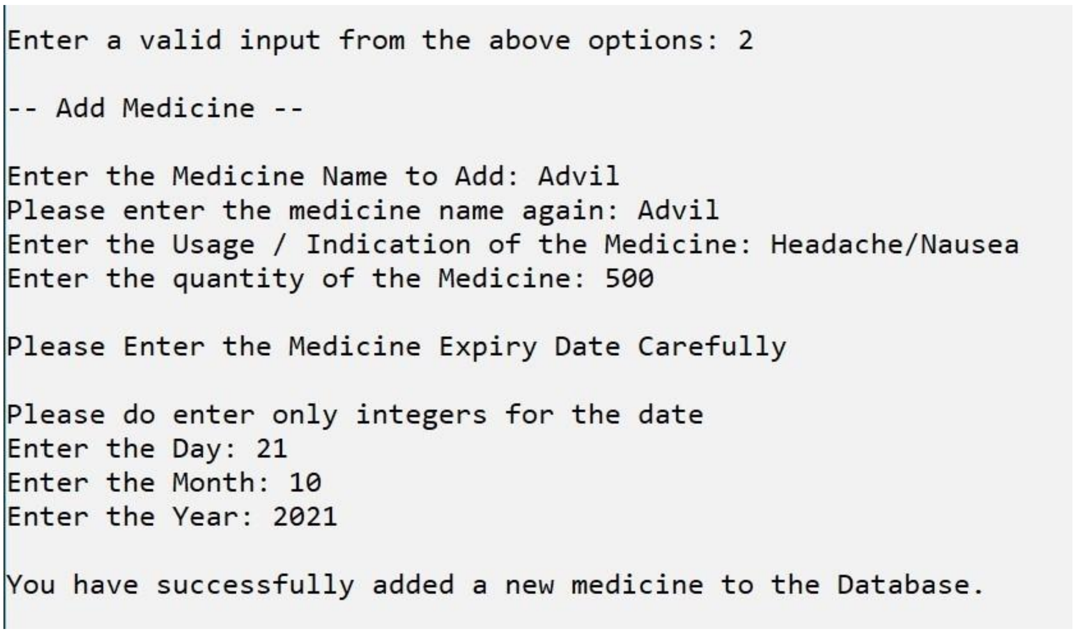
```
-------- Medicine Management Menu --------

Press (1) to see the Medicine List
Press (2) to Add a Medicine
Press (3) to See Expired Medicines
Press (4) to Update a Medicine
Press (5) to Delete a Expired Medicine
Press (6) to go to Admin Menu
Press (7) to Exit the Program

Enter a valid input from the above options:
```

Option 1 : see the medicine list



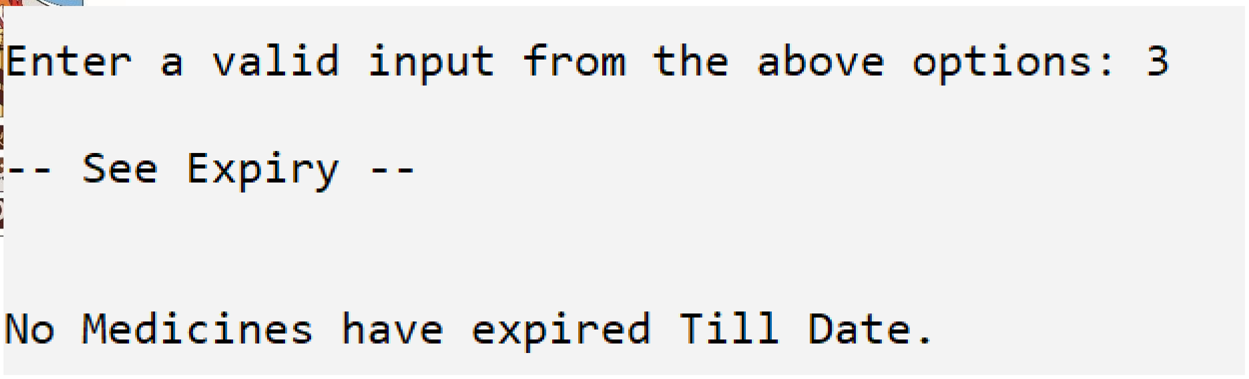
```
Enter a valid input from the above options: 1


+--------------------------------------+----------------------------------------------------+----------+-------------+
|            Medicine Name             |                 Usage / Indication                 | Quantity | Expiry Date |
+--------------------------------------+----------------------------------------------------+----------+-------------+
|      1.0 Catgut Suture Needles       |                      Suturing                      |    0     | 2021-10-05 |
|      2.0 Catgut Suture Needles       |                      Suturing                      |    0     | 2021-10-05 |
|         Acne Star Ointment           |                        Acne                        |    0     | 2021-10-05 |
|         Actowin nasal Drops          |                 Clear Nasal Pathway                |    0     | 2021-10-05 |
|           Acyclofenac - P            |               Ortho And Muscular Pain              |    0     | 2021-10-05 |
|          Acyclovir Ointment          |                  Herpes Infections                 |    0     | 2021-10-05 |
|              Arm Sling               |                    Arm Fracture                    |    0     | 2021-10-05 |
|           Asthalin Inhaler           |             Bronchial Asthma And Dypnea            |    0     | 2021-10-05 |
|          Asthalin Respules           |                    Expectorant                     |    0     | 2021-10-05 |
|          Bactogen Ointment           |                Bactericidal Solution               |    0     | 2021-10-05 |
|         Beclasone C Ointment         |                   Skin Infections                  |    0     | 2021-10-05 |
|          Benzac AC Ointment          |                   Skin Infections                  |    0     | 2021-10-05 |
|        Betadin Gargle Solution       |                   Ent Infections                   |    0     | 2021-10-05 |
|           Betadine solution          |                Bactericidal Solution               |    0     | 2021-10-05 |
|       Betamethosone Ointment         |                   Skin Infections                  |    0     | 2021-10-05 |
|           Betzee Ointment            |                   Skin Infections                  |    0     | 2021-10-05 |
```

Option 2 : adding a new medicine



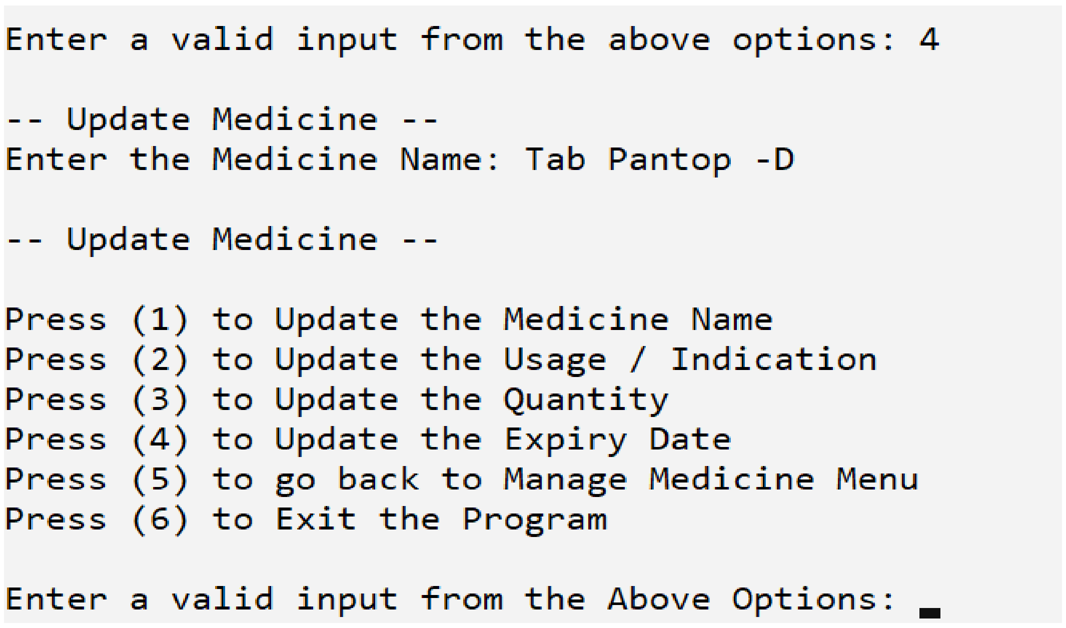
```
Enter a valid input from the above options: 2

-- Add Medicine --

Enter the Medicine Name to Add: Advil
Please enter the medicine name again: Advil
Enter the Usage / Indication of the Medicine: Headache/Nausea
Enter the quantity of the Medicine: 500

Please Enter the Medicine Expiry Date Carefully

Please do enter only integers for the date
Enter the Day: 21
Enter the Month: 10
Enter the Year: 2021

You have successfully added a new medicine to the Database.
```

Option 3 : to see the expired medicines



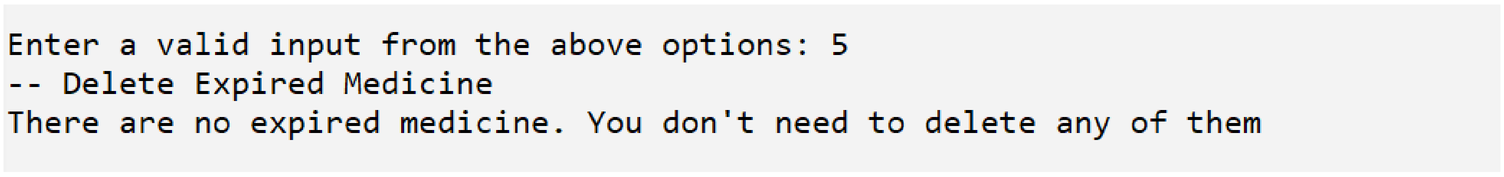
```
Enter a valid input from the above options: 3

-- See Expiry --


No Medicines have expired Till Date.
```

Option 4 : to update a medicine



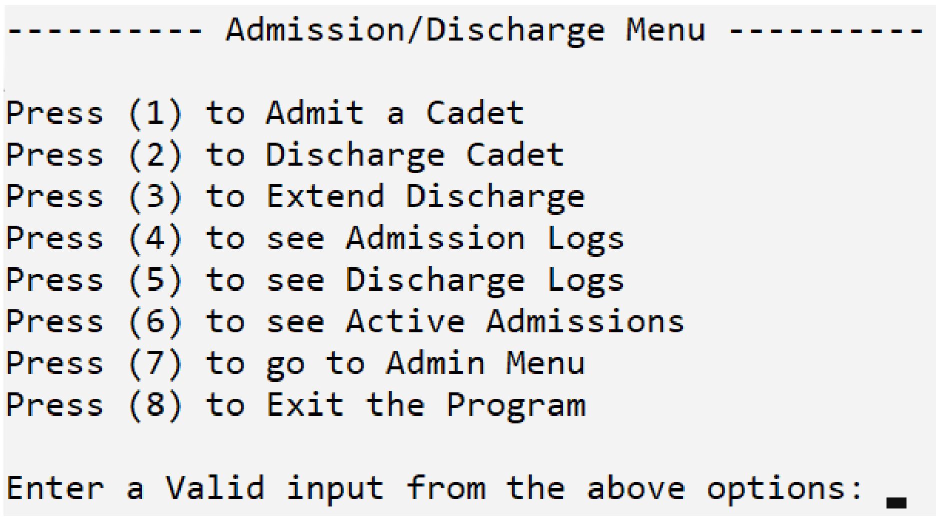
```
Enter a valid input from the above options: 4

-- Update Medicine --
Enter the Medicine Name: Tab Pantop -D

-- Update Medicine --

Press (1) to Update the Medicine Name
Press (2) to Update the Usage / Indication
Press (3) to Update the Quantity
Press (4) to Update the Expiry Date
Press (5) to go back to Manage Medicine Menu
Press (6) to Exit the Program

Enter a valid input from the Above Options: _
```

Option 5 : to delete an expired medicines



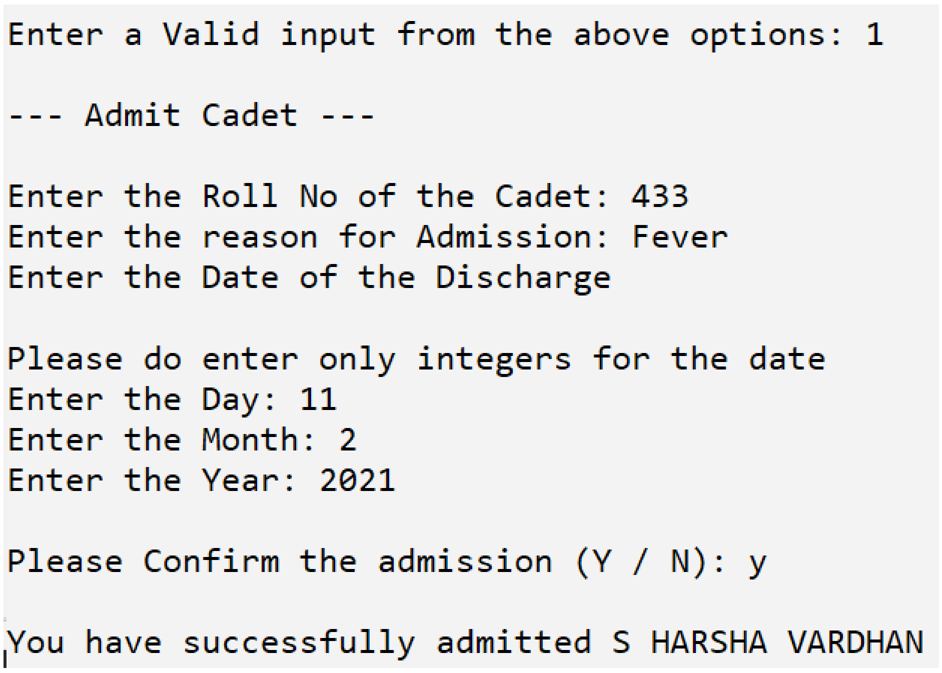
```
Enter a valid input from the above options: 5
-- Delete Expired Medicine
There are no expired medicine. You don't need to delete any of them
```

Admission/ Discharge Menu



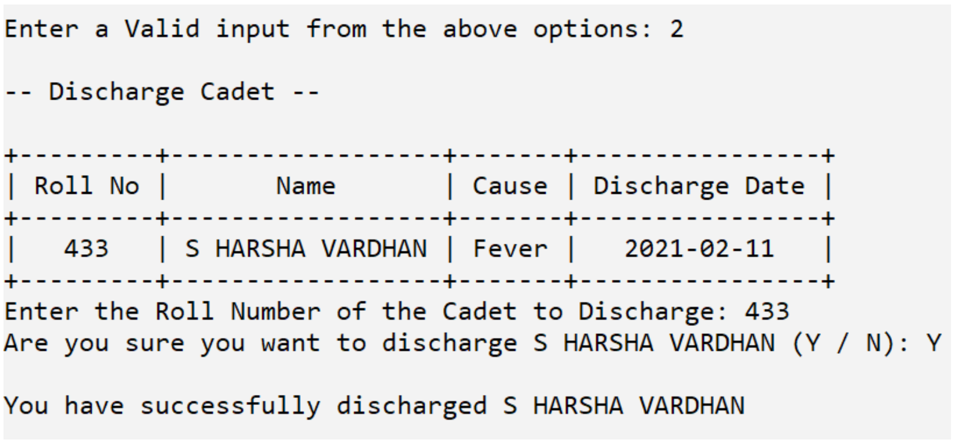
```
---------- Admission/Discharge Menu ----------

Press (1) to Admit a Cadet
Press (2) to Discharge Cadet
Press (3) to Extend Discharge
Press (4) to see Admission Logs
Press (5) to see Discharge Logs
Press (6) to see Active Admissions
Press (7) to go to Admin Menu
Press (8) to Exit the Program

Enter a Valid input from the above options:
```

Option 1 : to admit a cadet to the dispensary



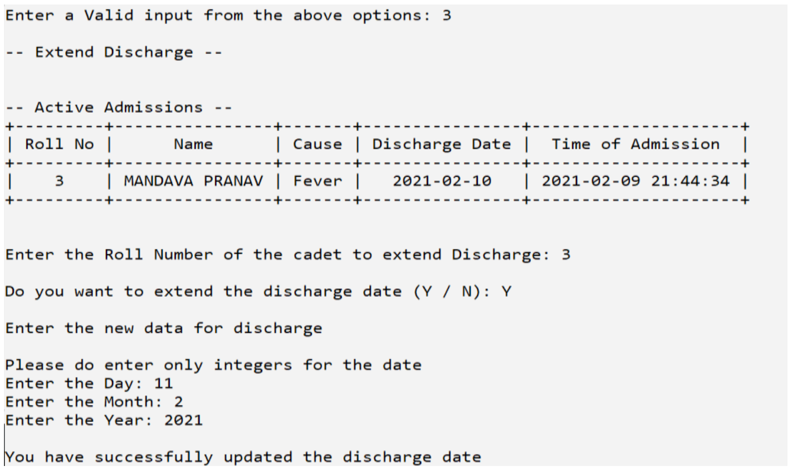
```
Enter a Valid input from the above options: 1

--- Admit Cadet ---

Enter the Roll No of the Cadet: 433
Enter the reason for Admission: Fever
Enter the Date of the Discharge

Please do enter only integers for the date
Enter the Day: 11
Enter the Month: 2
Enter the Year: 2021

Please Confirm the admission (Y / N): y

You have successfully admitted S HARSHA VARDHAN
```

Option 2 : discharge the cadet



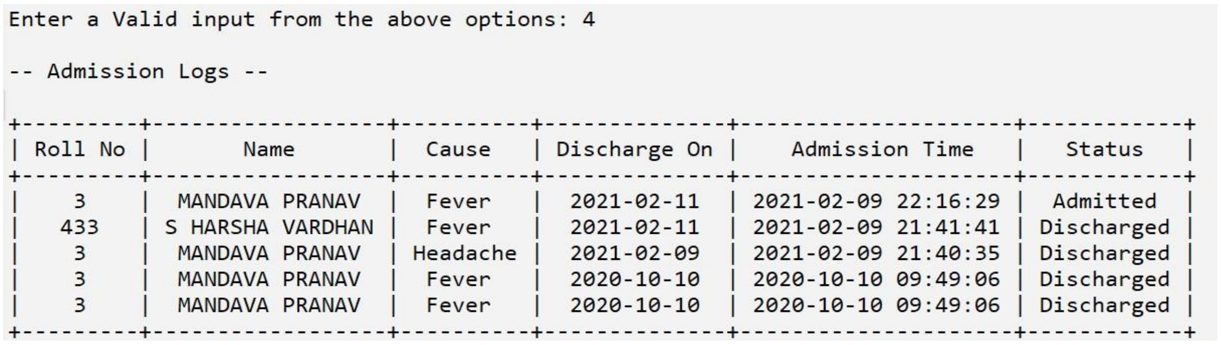
```
Enter a Valid input from the above options: 2

-- Discharge Cadet --

+---------+-----------------+-------+----------------+
| Roll No |       Name      | Cause | Discharge Date |
+---------+-----------------+-------+----------------+
|   433   | S HARSHA VARDHAN | Fever |   2021-02-11   |
+---------+-----------------+-------+----------------+
Enter the Roll Number of the Cadet to Discharge: 433
Are you sure you want to discharge S HARSHA VARDHAN (Y / N): Y

You have successfully discharged S HARSHA VARDHAN
```

Option 3 : to extend the discharge date of the cadet



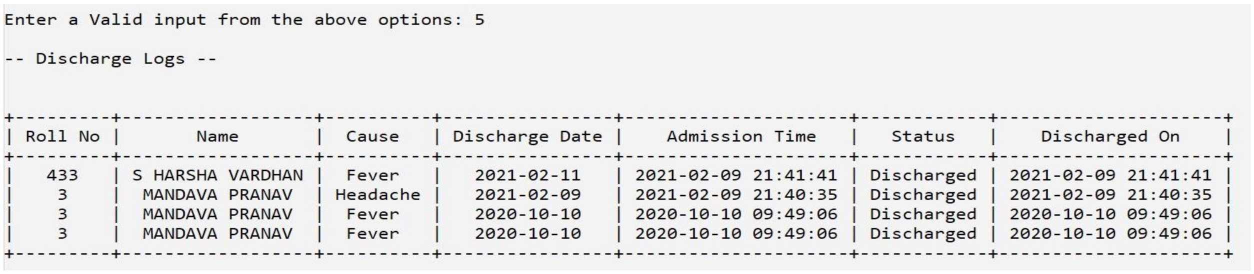
```
Enter a Valid input from the above options: 3

-- Extend Discharge --


-- Active Admissions --
+---------+----------------+-------+----------------+---------------------+
| Roll No |      Name      | Cause | Discharge Date |  Time of Admission  |
+---------+----------------+-------+----------------+---------------------+
|    3    | MANDAVA PRANAV | Fever |   2021-02-10   | 2021-02-09 21:44:34 |
+---------+----------------+-------+----------------+---------------------+


Enter the Roll Number of the cadet to extend Discharge: 3

Do you want to extend the discharge date (Y / N): Y

Enter the new data for discharge

Please do enter only integers for the date
Enter the Day: 11
Enter the Month: 2
Enter the Year: 2021

You have successfully updated the discharge date
```

Option 4 : see Admission Logs



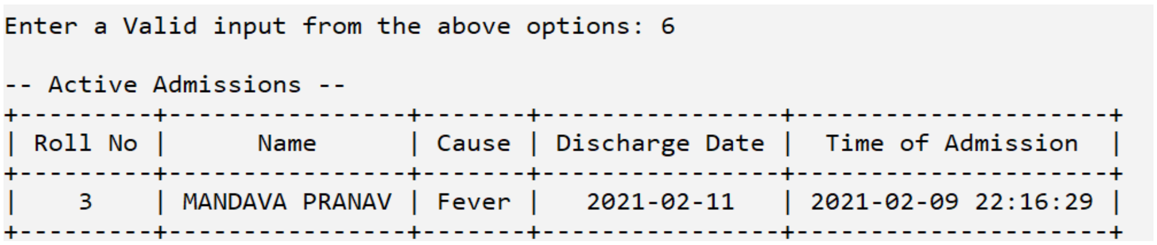
```
Enter a Valid input from the above options: 4

-- Admission Logs --


+---------+------------------+----------+--------------+---------------------+-----------+
| Roll No |       Name       |  Cause   | Discharge On |   Admission Time    |  Status   |
+---------+------------------+----------+--------------+---------------------+-----------+
|    3    |  MANDAVA PRANAV  |  Fever   |  2021-02-11  | 2021-02-09 22:16:29 |  Admitted |
|   433   | S HARSHA VARDHAN |  Fever   |  2021-02-11  | 2021-02-09 21:41:41 | Discharged|
|    3    |  MANDAVA PRANAV  | Headache |  2021-02-09  | 2021-02-09 21:40:35 | Discharged|
|    3    |  MANDAVA PRANAV  |  Fever   |  2020-10-10  | 2020-10-10 09:49:06 | Discharged|
|    3    |  MANDAVA PRANAV  |  Fever   |  2020-10-10  | 2020-10-10 09:49:06 | Discharged|
+---------+------------------+----------+--------------+---------------------+-----------+
```

Option 5 : see discharge Logs



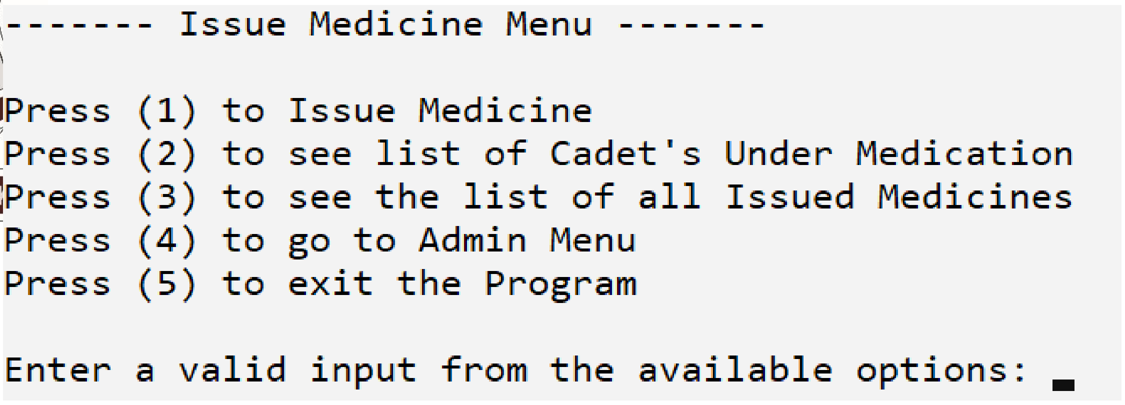
```
Enter a Valid input from the above options: 5

-- Discharge Logs --


+---------+------------------+----------+----------------+---------------------+------------+---------------------+
| Roll No |       Name       |  Cause   | Discharge Date |   Admission Time    |   Status   |    Discharged On    |
+---------+------------------+----------+----------------+---------------------+------------+---------------------+
|   433   | S HARSHA VARDHAN |  Fever   |   2021-02-11   | 2021-02-09 21:41:41 | Discharged | 2021-02-09 21:41:41 |
|    3    |  MANDAVA PRANAV  | Headache |   2021-02-09   | 2021-02-09 21:40:35 | Discharged | 2021-02-09 21:40:35 |
|    3    |  MANDAVA PRANAV  |  Fever   |   2020-10-10   | 2020-10-10 09:49:06 | Discharged | 2020-10-10 09:49:06 |
|    3    |  MANDAVA PRANAV  |  Fever   |   2020-10-10   | 2020-10-10 09:49:06 | Discharged | 2020-10-10 09:49:06 |
+---------+------------------+----------+----------------+---------------------+------------+---------------------+
```

Option 6 : see Active Admissions



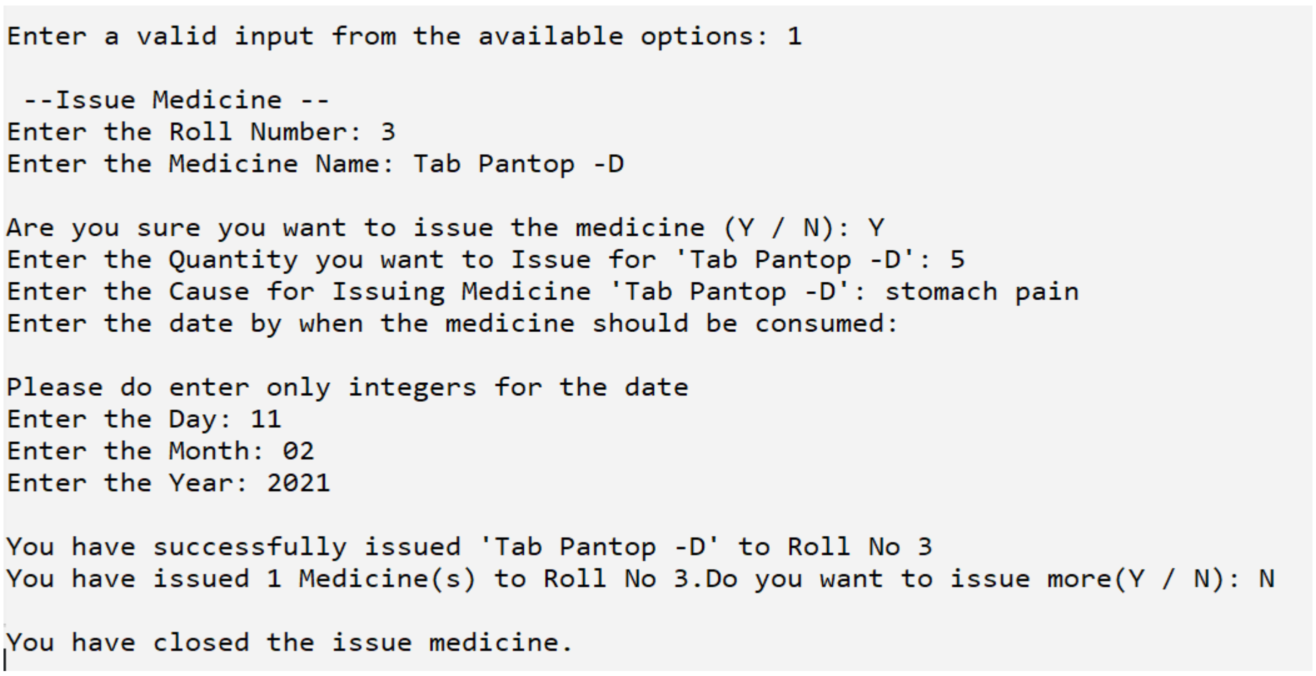
```
Enter a Valid input from the above options: 6

-- Active Admissions --
+---------+----------------+-------+----------------+---------------------+
| Roll No |      Name      | Cause | Discharge Date |  Time of Admission  |
+---------+----------------+-------+----------------+---------------------+
|    3    | MANDAVA PRANAV | Fever |   2021-02-11   | 2021-02-09 22:16:29 |
+---------+----------------+-------+----------------+---------------------+
```

Issue Medicine Menu



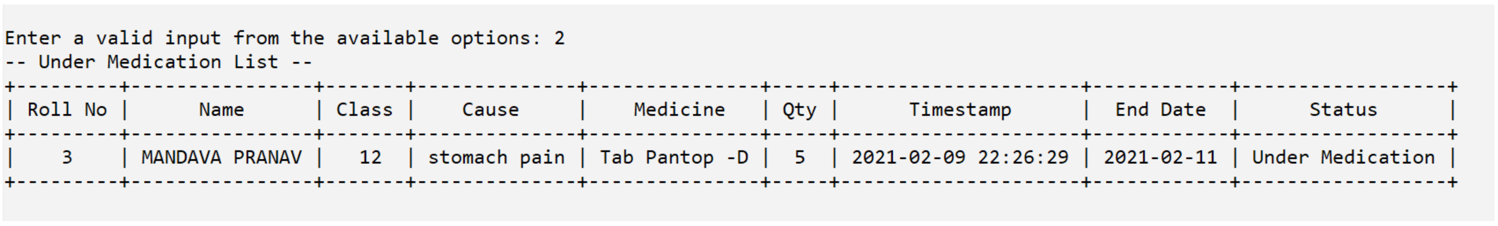
```
------- Issue Medicine Menu -------

Press (1) to Issue Medicine
Press (2) to see list of Cadet's Under Medication
Press (3) to see the list of all Issued Medicines
Press (4) to go to Admin Menu
Press (5) to exit the Program

Enter a valid input from the available options: _
```

Option 1 : to Issue a Medicine to the cadet



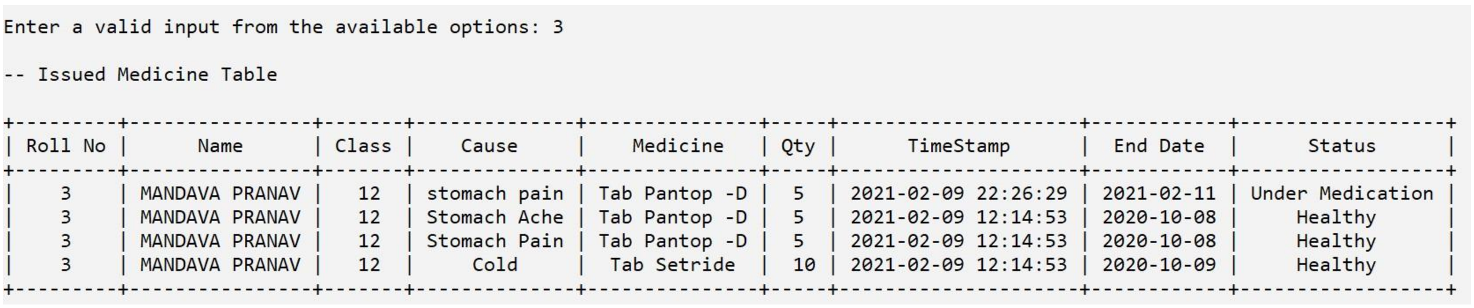
```
Enter a valid input from the available options: 1

 --Issue Medicine --
Enter the Roll Number: 3
Enter the Medicine Name: Tab Pantop -D

Are you sure you want to issue the medicine (Y / N): Y
Enter the Quantity you want to Issue for 'Tab Pantop -D': 5
Enter the Cause for Issuing Medicine 'Tab Pantop -D': stomach pain
Enter the date by when the medicine should be consumed:

Please do enter only integers for the date
Enter the Day: 11
Enter the Month: 02
Enter the Year: 2021

You have successfully issued 'Tab Pantop -D' to Roll No 3
You have issued 1 Medicine(s) to Roll No 3.Do you want to issue more(Y / N): N

You have closed the issue medicine.
```

Option 2 : to see the list of cadets under medication



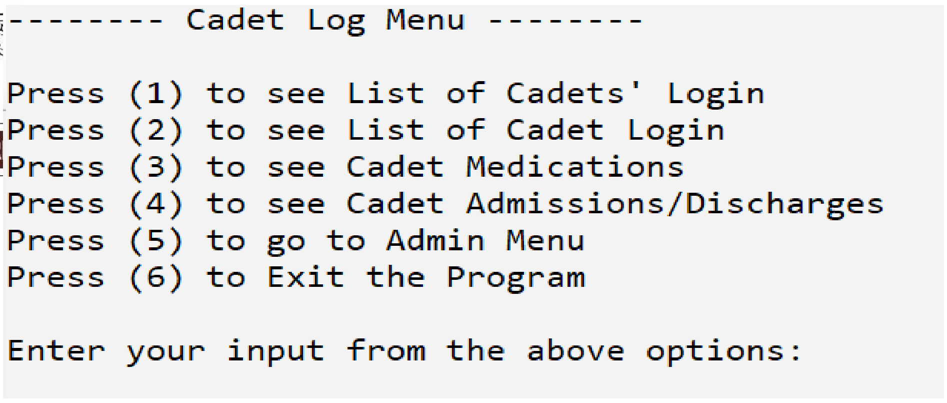
```
Enter a valid input from the available options: 2
-- Under Medication List --
+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
| Roll No |      Name     | Class |    Cause     |   Medicine    | Qty |      Timestamp      |  End Date  |     Status      |
+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
|    3    | MANDAVA PRANAV |  12   | stomach pain | Tab Pantop -D |  5  | 2021-02-09 22:26:29 | 2021-02-11 | Under Medication |
+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
```

Option 3 : to see the list of all Issued Medicines

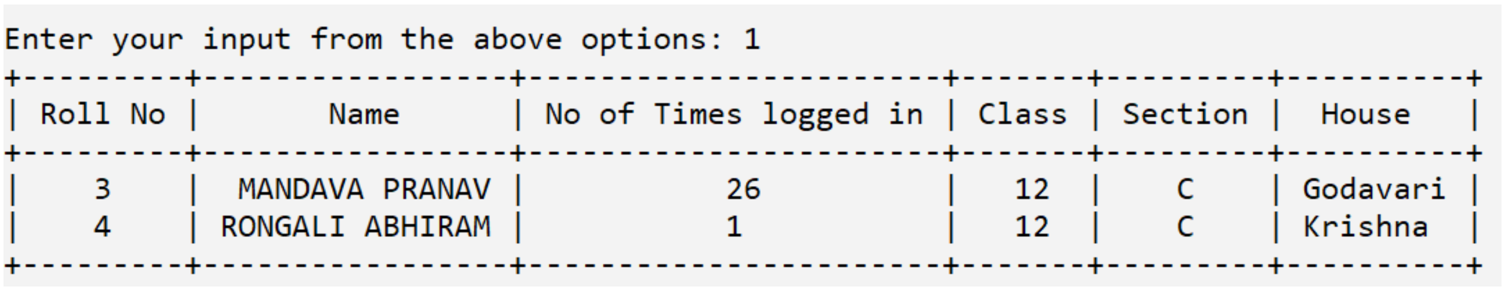


```
Enter a valid input from the available options: 3

-- Issued Medicine Table

+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
| Roll No |      Name     | Class |    Cause     |   Medicine    | Qty |      TimeStamp      |  End Date  |     Status      |
+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
|    3    | MANDAVA PRANAV |  12   | stomach pain | Tab Pantop -D |  5  | 2021-02-09 22:26:29 | 2021-02-11 | Under Medication |
|    3    | MANDAVA PRANAV |  12   | Stomach Ache | Tab Pantop -D |  5  | 2021-02-09 12:14:53 | 2020-10-08 |     Healthy     |
|    3    | MANDAVA PRANAV |  12   | Stomach Pain | Tab Pantop -D |  5  | 2021-02-09 12:14:53 | 2020-10-08 |     Healthy     |
|    3    | MANDAVA PRANAV |  12   |     Cold     |  Tab Setride  | 10  | 2021-02-09 12:14:53 | 2020-10-09 |     Healthy     |
+---------+---------------+-------+--------------+---------------+-----+---------------------+------------+-----------------+
```

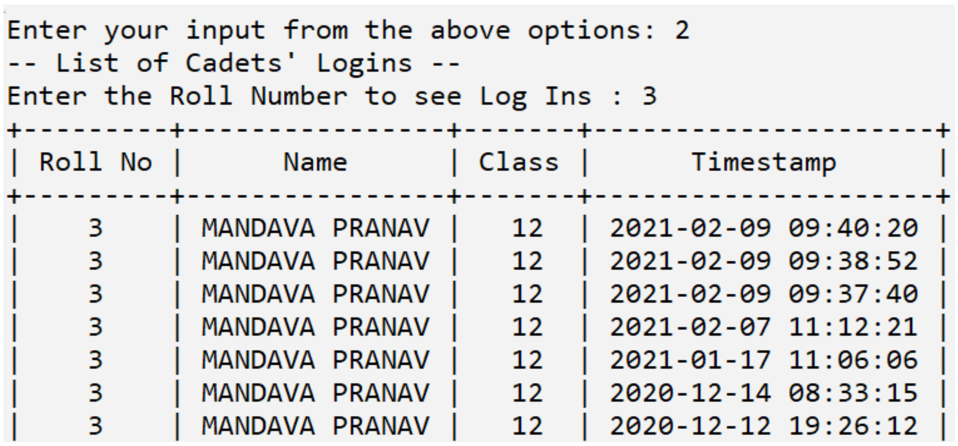
Cadet Log Menu


```
-------- Cadet Log Menu --------

Press (1) to see List of Cadets' Login
Press (2) to see List of Cadet Login
Press (3) to see Cadet Medications
Press (4) to see Cadet Admissions/Discharges
Press (5) to go to Admin Menu
Press (6) to Exit the Program

Enter your input from the above options:
```

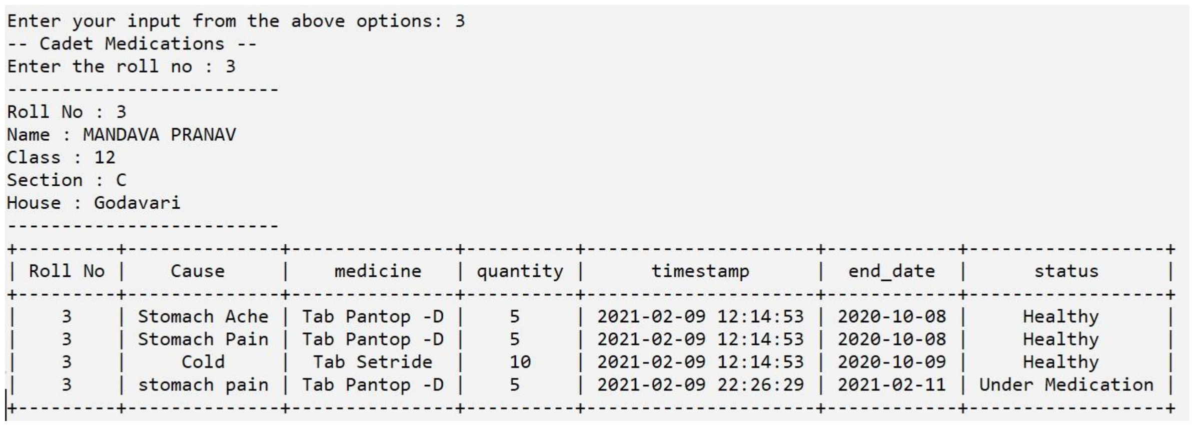
Option 1 : to list of all the cadets login


```
Enter your input from the above options: 1
+---------+-----------------+----------------------+-------+---------+----------+
| Roll No |       Name      | No of Times logged in | Class | Section |   House  |
+---------+-----------------+----------------------+-------+---------+----------+
|    3    |  MANDAVA PRANAV |          26          |   12  |    C    | Godavari |
|    4    | RONGALI ABHIRAM |          1           |   12  |    C    | Krishna  |
+---------+-----------------+----------------------+-------+---------+----------+
```

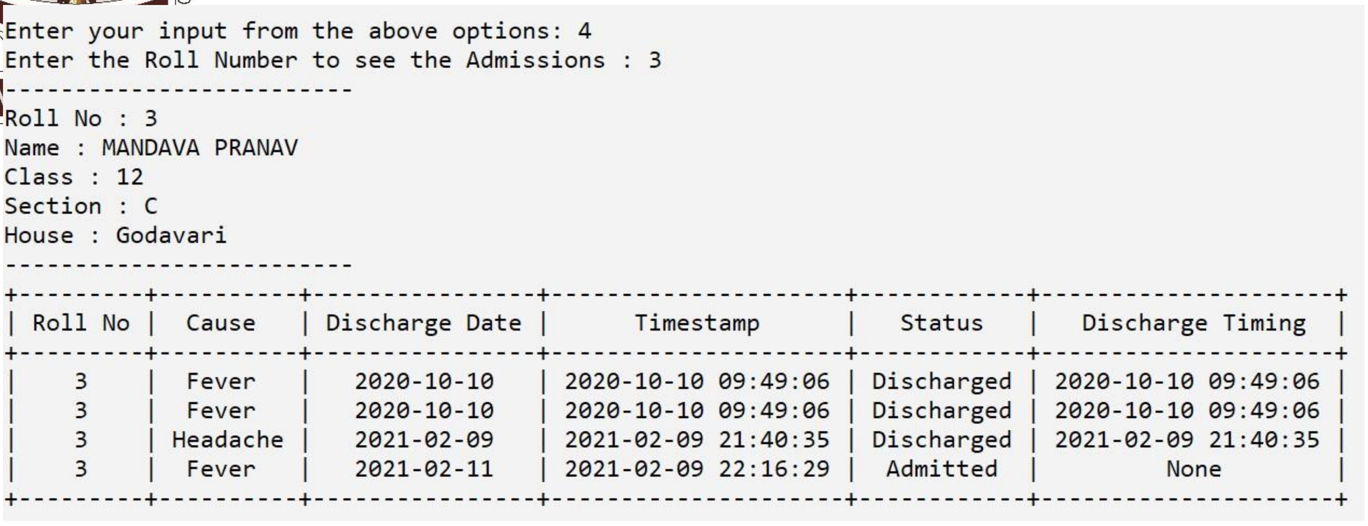
Option 2 : to see list of a specific cadet login


```
Enter your input from the above options: 2
-- List of Cadets' Logins --
Enter the Roll Number to see Log Ins : 3
+---------+----------------+-------+---------------------+
| Roll No |      Name      | Class |      Timestamp      |
+---------+----------------+-------+---------------------+
|    3    | MANDAVA PRANAV |   12  | 2021-02-09 09:40:20 |
|    3    | MANDAVA PRANAV |   12  | 2021-02-09 09:38:52 |
|    3    | MANDAVA PRANAV |   12  | 2021-02-09 09:37:40 |
|    3    | MANDAVA PRANAV |   12  | 2021-02-07 11:12:21 |
|    3    | MANDAVA PRANAV |   12  | 2021-01-17 11:06:06 |
|    3    | MANDAVA PRANAV |   12  | 2020-12-14 08:33:15 |
|    3    | MANDAVA PRANAV |   12  | 2020-12-12 19:26:12 |
```

Option 3 : to see cadet Medications

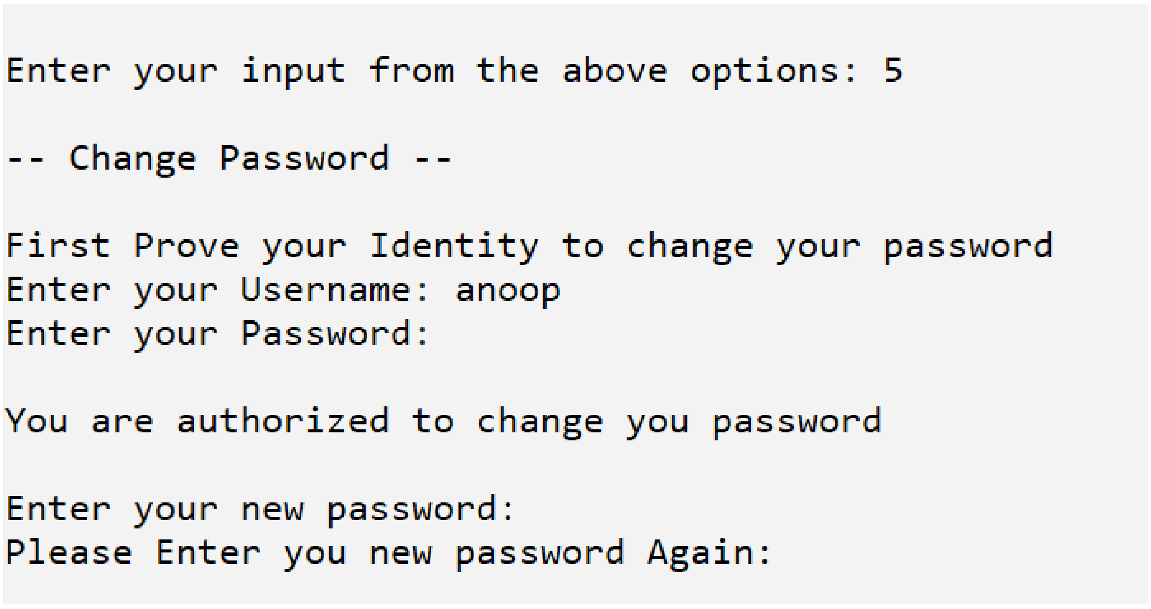

```
Enter your input from the above options: 3
-- Cadet Medications --
Enter the roll no : 3
------------------------
Roll No : 3
Name : MANDAVA PRANAV
Class : 12
Section : C
House : Godavari
------------------------
+---------+--------------+---------------+----------+---------------------+------------+-----------------+
| Roll No |    Cause     |    medicine   | quantity |      timestamp      |  end_date  |      status     |
+---------+--------------+---------------+----------+---------------------+------------+-----------------+
|    3    | Stomach Ache | Tab Pantop -D |    5     | 2021-02-09 12:14:53 | 2020-10-08 |     Healthy     |
|    3    | Stomach Pain | Tab Pantop -D |    5     | 2021-02-09 12:14:53 | 2020-10-08 |     Healthy     |
|    3    |     Cold     |  Tab Setride  |    10    | 2021-02-09 12:14:53 | 2020-10-09 |     Healthy     |
|    3    | stomach pain | Tab Pantop -D |    5     | 2021-02-09 22:26:29 | 2021-02-11 | Under Medication |
+---------+--------------+---------------+----------+---------------------+------------+-----------------+
```

Option 4 : to see a cadet’s admissions & discharges



```
Enter your input from the above options: 4
Enter the Roll Number to see the Admissions : 3
-------------------------
Roll No : 3
Name : MANDAVA PRANAV
Class : 12
Section : C
House : Godavari
-------------------------
+---------+---------+--------------+--------------------+------------+---------------------+
| Roll No |  Cause  | Discharge Date |      Timestamp      |   Status   |  Discharge Timing   |
+---------+---------+--------------+--------------------+------------+---------------------+
|    3    |  Fever  |   2020-10-10   | 2020-10-10 09:49:06 | Discharged | 2020-10-10 09:49:06 |
|    3    |  Fever  |   2020-10-10   | 2020-10-10 09:49:06 | Discharged | 2020-10-10 09:49:06 |
|    3    | Headache |   2021-02-09   | 2021-02-09 21:40:35 | Discharged | 2021-02-09 21:40:35 |
|    3    |  Fever  |   2021-02-11   | 2021-02-09 22:16:29 |  Admitted  |         None        |
+---------+---------+--------------+--------------------+------------+---------------------+
```

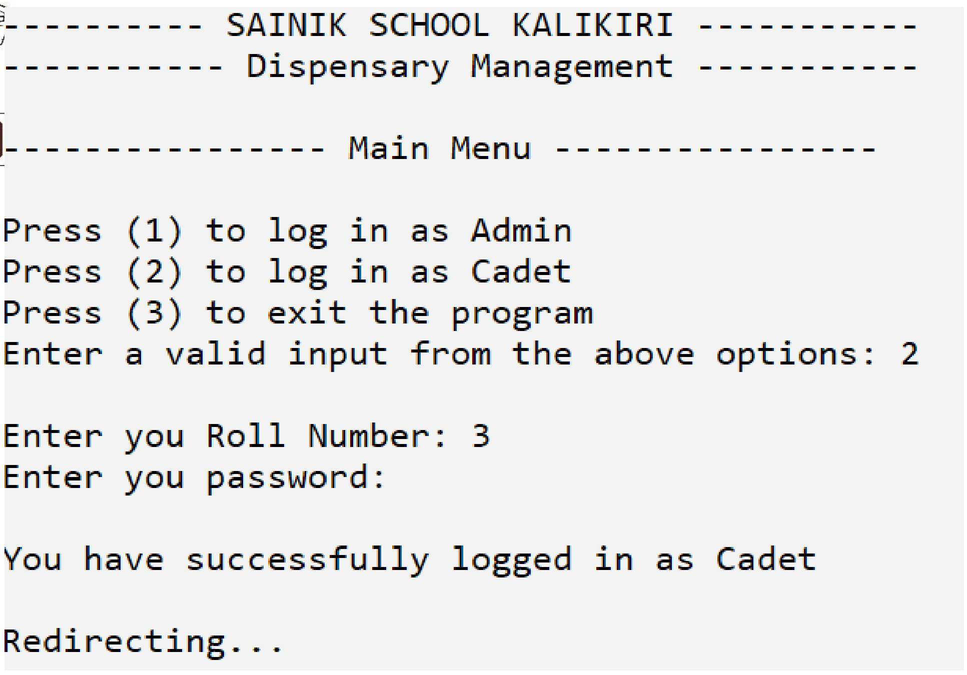
Admin – Change your password



```
Enter your input from the above options: 5

-- Change Password --

First Prove your Identity to change your password
Enter your Username: anoop
Enter your Password:

You are authorized to change you password

Enter your new password:
Please Enter you new password Again:
```

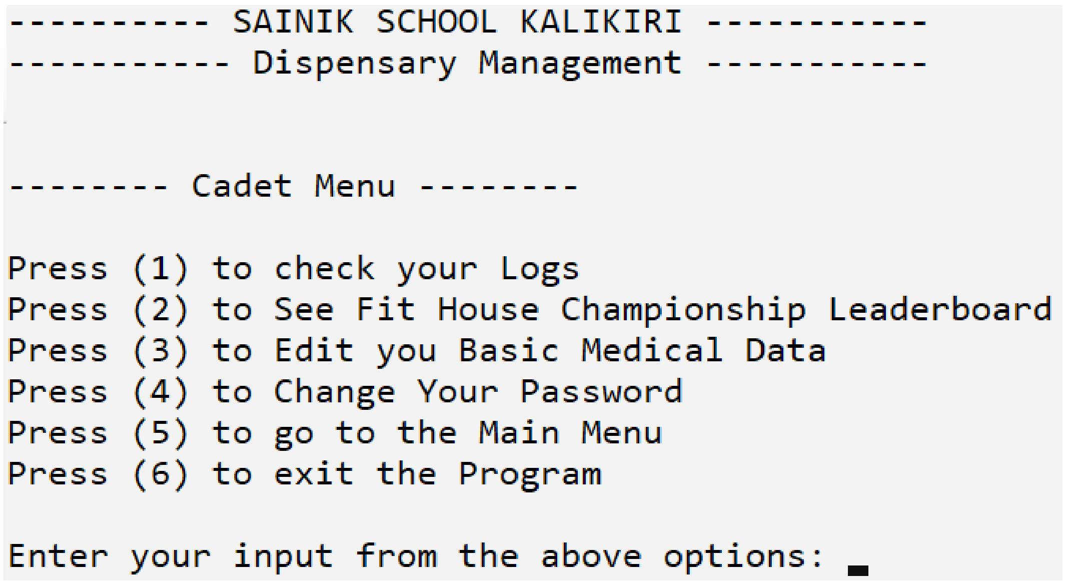
Note : To maintain privacy of the user the password won’t be shown the computer screen at any point of time

Logging in as Cadet



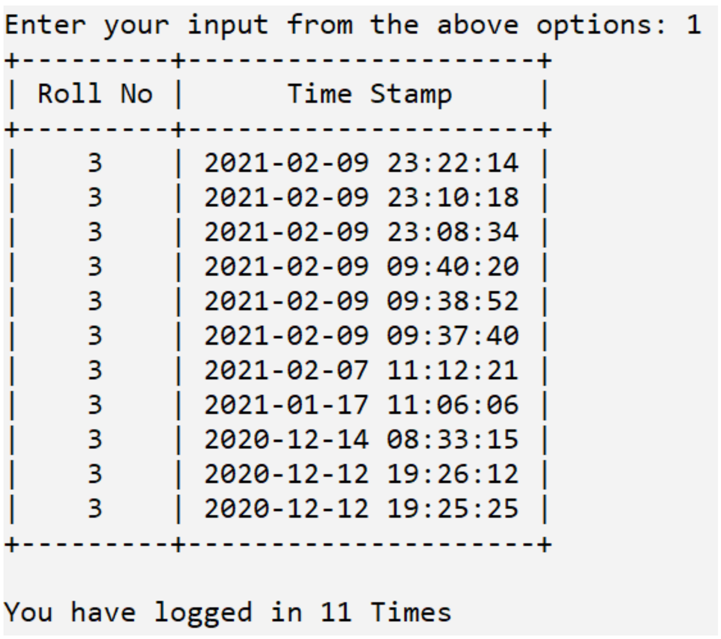
```
---------- SAINIK SCHOOL KALIKIRI -----------
----------- Dispensary Management -----------

---------------- Main Menu ----------------

Press (1) to log in as Admin
Press (2) to log in as Cadet
Press (3) to exit the program
Enter a valid input from the above options: 2

Enter you Roll Number: 3
Enter you password:

You have successfully logged in as Cadet

Redirecting...
```

Main Menu of the Cadet



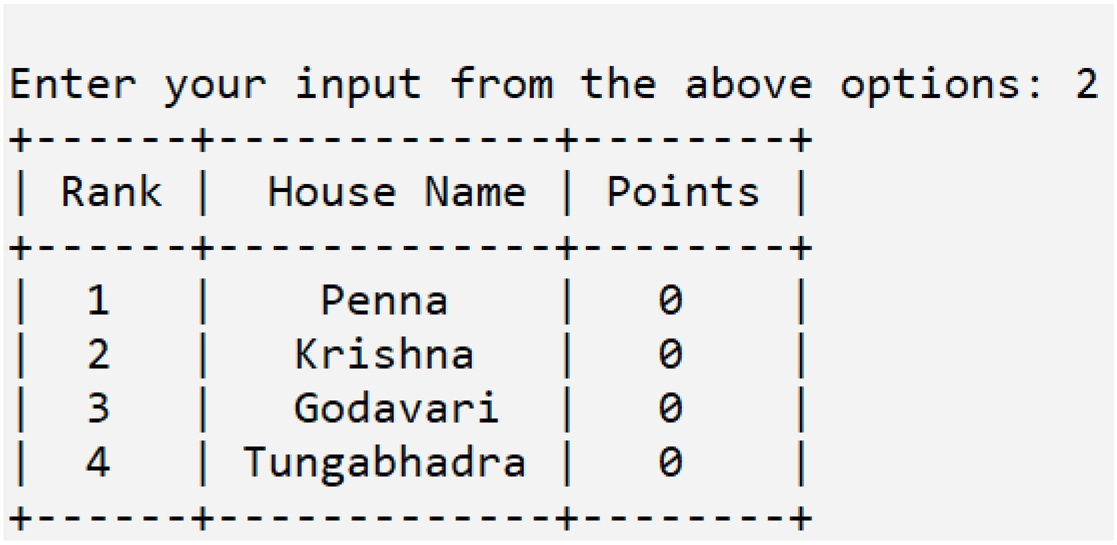
```
---------- SAINIK SCHOOL KALIKIRI -----------
----------- Dispensary Management -----------


-------- Cadet Menu --------

Press (1) to check your Logs
Press (2) to See Fit House Championship Leaderboard
Press (3) to Edit you Basic Medical Data
Press (4) to Change Your Password
Press (5) to go to the Main Menu
Press (6) to exit the Program

Enter your input from the above options:
```

Option 1 : to check cadet logs



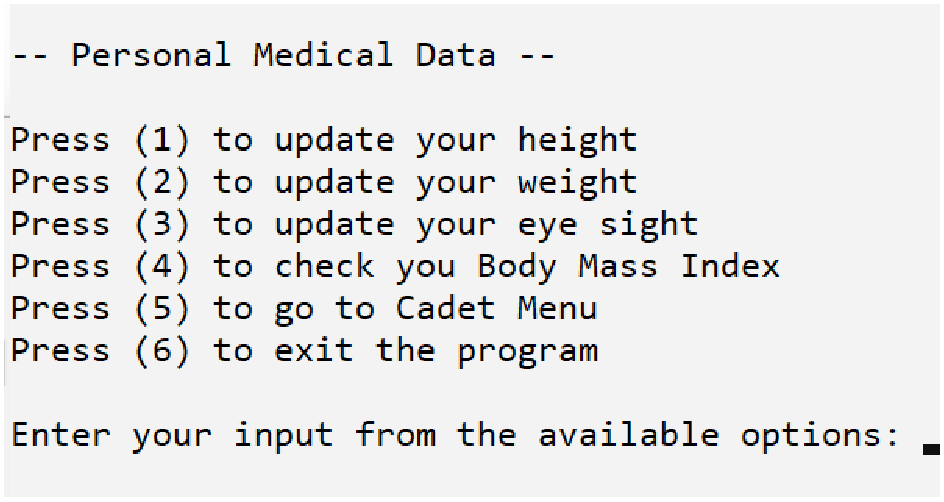
```
Enter your input from the above options: 1
+---------+---------------------+
| Roll No |      Time Stamp     |
+---------+---------------------+
|    3    | 2021-02-09 23:22:14 |
|    3    | 2021-02-09 23:10:18 |
|    3    | 2021-02-09 23:08:34 |
|    3    | 2021-02-09 09:40:20 |
|    3    | 2021-02-09 09:38:52 |
|    3    | 2021-02-09 09:37:40 |
|    3    | 2021-02-07 11:12:21 |
|    3    | 2021-01-17 11:06:06 |
|    3    | 2020-12-14 08:33:15 |
|    3    | 2020-12-12 19:26:12 |
|    3    | 2020-12-12 19:25:25 |
+---------+---------------------+

You have logged in 11 Times
```

Option 2 : to See Fit House Championship Leaderboard



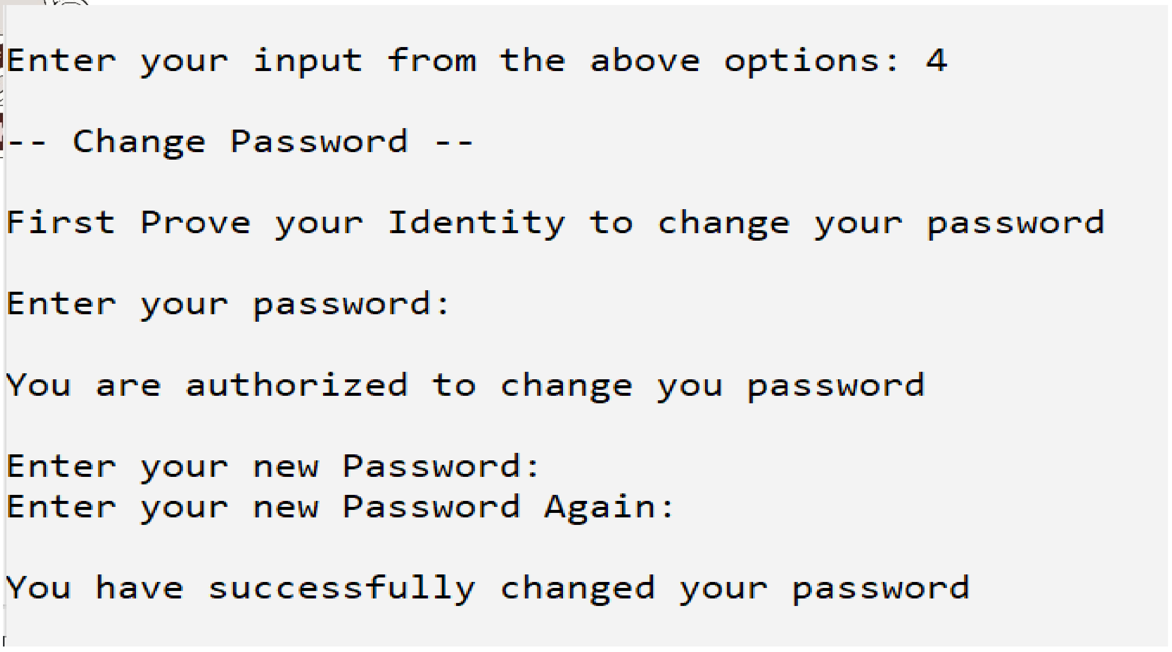
```
Enter your input from the above options: 2
+------+-------------+--------+
| Rank |  House Name | Points |
+------+-------------+--------+
|  1   |    Penna    |   0    |
|  2   |   Krishna   |   0    |
|  3   |   Godavari  |   0    |
|  4   | Tungabhadra |   0    |
+------+-------------+--------+
```

Option 3 : to Edit the cadet's Basic Medical Data



```
-- Personal Medical Data --

Press (1) to update your height
Press (2) to update your weight
Press (3) to update your eye sight
Press (4) to check you Body Mass Index
Press (5) to go to Cadet Menu
Press (6) to exit the program

Enter your input from the available options:
```

Option 4 : to change cadet’s password



```
Enter your input from the above options: 4

-- Change Password --

First Prove your Identity to change your password

Enter your password:

You are authorized to change you password

Enter your new Password:
Enter your new Password Again:

You have successfully changed your password
```

4. BIBLIOGRAPHY

1. Computer Science with Python [Textbook XII] by Sumita Arora
2. https://ptable.readthedocs.io/en/latest/
3. https://docs.python.org/3/library/getpass.html