**DSA Project Report**

**Project Title:** Pharmacy Management System

**Project Members: -**

* Gull Rukh Khan 11075
* Usman Ahmed 10584

**Introduction: -**

The Pharmacy Management System is a project developed to automate medical stores’ activities and improve their productivity. This helps pharmacies organize, manage, and secure drug information efficiently. Its’ features aids in the resolution of challenges with manual pharmacy management.

A Pharmacy Management System can also help you keep track of your drug supplies. Prescriptions are proper and supplied in precise amounts using Pharmacy Management software. It oversees and manages the pharmacy team to preserve strong working relationships and outcomes. This can also improve quality and customer satisfaction ratings, as well as keep medicines from going bad.

**Data Structure used: -**

* **Stack:**

A stack is a linear data structure that stores items in a Last-In/First-Out (LIFO) or First-In/Last-Out (FILO) manner. In stack, a new element is added at one end and an element is removed from that end only. The insert and delete operations are often called push and pop.

* **Linked-list:**

Linked Lists are a data structure that store data in the form of a chain. The structure of a linked list is such that each piece of data has a connection to the next one (and sometimes the previous data as well). Each element in a linked list is called a node.

**Code: -**

import datetime

import os

import random

import threading

# First Block We Have Searching:

# Block 1

#Two type Of searching We Perform.

def binary\_search(arr, low, high, x):

# Check base case

if high >= low:

mid = (high + low) // 2

# If element is present at the middle itself

if int(arr[mid]) == int(x):

return arr[mid]

# If element is smaller than mid, then it can only

# be present in left subarray

elif int(arr[mid]) > int(x):

return binary\_search(arr, low, mid - 1, x)

# Else the element can only be present in right subarray

else:

return binary\_search(arr, mid + 1, high, x)

else:

# Element is not present in the array

return -1

# Then we hhave linear search

def linearsearch(arr, x):

for i in range(len(arr)):

if str(arr[i]) == str(x):

return i

else:

return -1

#print()

#Block 2

#we have Block of Sorting

#Two type of sorting

def insertion\_sort(arr):

# move to 2 element find whether its big or not if its bigger than swap first element with 2

for step in range(1, len(arr)):

key = arr[step]

j = step - 1

while j >= 0 and key > arr[j]:

arr[j + 1] = arr[j]

j = j - 1

arr[j + 1] = key

# than we have Bubble\_Sort

def bubblesort(arr):

#Bubble move 1 by one to another account

n = len(arr)

for i in range(n - 1):

for j in range(0, n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

#Now we use another dsa which is linked list

#in link list we perform everything which is store in file==""medicine.txt"

#features we have in linklist are

#Every data comes from file medicines.txt

#Buy any medicine = Purchase Medicines shown at compile time

#Add Any Medicines in file medicines.txt save in pythonproject

#Edit Any Medicines in file data

#Search Any Medicines

#Delete Any Medicines

#DeleteAll

#Display

#exit

class Node:

def \_\_init\_\_(self, particular, qty, unitprice):

self.particular = particular

self.qty = qty

self.unitprice = unitprice

self.amount = int(self.qty) \* int(self.unitprice)

self.next = None

class MedicinesList:

def \_\_init\_\_(self):

self.Name = None

self.company = None

self.date = None

self.contact = None

self.totalbill = 0

self.balance = 0

self.start = None

def insert(self, particular, qty, unitprice):

if (self.start is None):

self.start = Node(particular, qty, unitprice)

else:

ptr = self.start

while (ptr.next != None):

ptr = ptr.next

ptr.next = Node(particular, qty, unitprice)

def createinvoice(self):

self.date = datetime.date.today()

check = True

while (check):

print()

print()

print()

any\_var = str(input('\t\tEnter Medicine ID: '))

qty = str(input('\t\tEnter Quantity: '))

a = int(input("\t\tEnter Your Balance Again "))

self.balance = a

print()

print()

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if (data[0] == any\_var):

particular = data[1]

unitprice = data[2]

self.insert(particular, qty, unitprice)

self.totalbill += int(qty) \* int(unitprice)

self.Balance = a - self.totalbill

inpt = input("Press E to End: ")

if (inpt == "E" or inpt == "e"):

check = False

if(a> self.totalbill):

print("\n\nYour Total Bill amount is " + "Total: " + str(self.totalbill))

print("\n\nYour Remainig balance is " + "Total: " + str(self.Balance))

else:

print("\n\nYour Total Bill amount is " + "Total: " + str(self.totalbill))

def Print(self):

if(self.balance > self.totalbill):

self.billno = random.randint(0, 100)

print()

print("||Bill: "

+ str(self.billno) +

"\t\t\t""||Date: "

+ str(self.date))

print()

ptr = self.start

i = 1

while (ptr != None):

print(str(i) + " \n" + ptr.particular +

" \nQuantity\n " + ptr.qty +

" \nPrice\n"

+ ptr.unitprice

+ " \nTotal Amount\n " + str(

ptr.amount))

ptr = ptr.next

i += 1

else:

print("You dont have enough Balance")

# We have another Block for performing in Medicines file whcih is save in pythonproject folder

# We have Linked list name MedicinesList.

class Medicines:

#in that class we add and delete in medicines.txt

medicines = []

def \_\_init\_\_(self, medid, medicinename, price, quantity):

self.medicineid = medid

self.medicinename = medicinename

self.price = price

self.quantity = quantity

def PurchaseMedicine(self):

#in that method data hmre ps addmedicine se ata hai yani agr ap kuch buy krna chate hai toh apko medicnices agr empty hai tw nhihoga

#Call That LinkedList here To perform every function

inputFile = open("medicines.txt", 'r')

lineList = inputFile.readlines()

print("\n",lineList)

Bill = MedicinesList()

Bill.createinvoice()

Bill.Print()

def medicinedetails(self):

file = "medicines.txt"

if os.path.exists(file):

for line in open(file, "r").readlines():

data = line.split(',')

self.medicines.append((data[0]))

def FileDeletion(self, file, file2):

#ye method deletion k lye use hua hai

#txt file mai id dalne se remove hojyega sb

with open(file, "r") as f:

with open(file2, "w+") as f1:

for line in f:

f1.write(line)

f.close()

f1.close()

if os.path.exists(file):

os.remove(file)

else:

print("The file does not exist")

def DisplayAllMedicine(self):

#txt mai display krne k lye

#txt file mai jo hoga id ki mdd se display hoga

self.medicines = []

self.medicinedetails()

print()

print()

order = int(input("\t1. Ascending Order\n\t2. Descending Order\n"))

print()

print()

if (order == 1):

t3 = threading.Thread(target=bubblesort(self.medicines), args=(10,))

t3.start()

t3.join()

if (os.path.exists("medicines.txt")):

for i in range(len(self.medicines)):

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if (self.medicines[i] == data[0]):

print()

print()

print(str("\n\tMedicine #") + str(i + 1))

print("\tID: \t" + str(data[0]))

print("\tName: \t" + str(data[1]))

print("\tPrice: \t" + str(data[2]))

print("\tQuantity: \t" + str(data[3]))

print()

print()

else:

print("xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx")

print("\tNo Medicines in Our Pharmacy")

elif (order == 2):

t4 = threading.Thread(target=bubblesort(self.medicines), args=(10,))

t4.start()

t4.join()

if (os.path.exists("medicines.txt")):

for i in range(len(self.medicines)):

# Stack for Descending Sort

any\_var = self.medicines.pop()

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if (any\_var == data[0]):

print(str("\n\tMedicine #") + str(i + 1))

print("\tID: \t" + str(data[0]))

print("\tName: \t" + str(data[1]))

print("\tPrice: \t" + str(data[2]))

print("\tQuantity: \t" + str(data[3]))

else:

print("\tNo Medicines in Our Pharmacy")

str(input("\tPress Any Key"))

def DisplayAvailableMedicines(self):

#method work nhi kr rha

#pending

self.medicines = []

self.medicinedetails()

order = int(input("\t1. Ascending Order\n\t2. Descending Order"))

if (order == 2):

# Sort

t5 = threading.Thread(target=insertion\_sort(self.medicines), args=(10,))

t5.start()

t5.join()

if (os.path.exists("medicines.txt")):

for i in range(len(self.medicines)):

# Queue

any\_var = self.medicines.pop(0)

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if ((any\_var == data[0]) and (int(data[3]) > 0)):

print(str("\tMedicine #") + str(i + 1))

print("\tID: \t" + str(data[0]))

print("\tName: \t" + str(data[1]))

print("\tPrice: \t" + str(data[2]))

print("\tQuantity: \t" + str(data[3]))

else:

print("\tNo Medicines in Our Pharmacy")

elif (order == 1):

t6 = threading.Thread(target=insertion\_sort(self.medicines), args=(10,))

t6.start()

t6.join()

if (os.path.exists("medicines.txt")):

for i in range(len(self.medicines)):

# Stack

any\_var = self.medicines.pop()

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if ((any\_var == data[0]) and (int(data[3]) > 0)):

print()

print(str("\tMedicine #") + str(i + 1))

print("\tID: \t" + str(data[0]))

print("\tName: \t" + str(data[1]))

print("\tPrice: \t" + str(data[2]))

print("\tQuantity: \t" + str(data[3]))

else:

print("\tNo Medicines in Our Pharmacy")

str(input("\tEnter Any Key To Continue "))

def SearchMedicine(self):

# txt file mai jo kuch hoga

# wo search krega id kimafdad se

# ye

var = True

for line in open("medicines.txt", "r").readlines():

data = line.split(',')

if (self.medicineid == data[0]):

print("\tID: " + str(data[0]))

print("\tName: " + str(data[1]))

print("\tPrice: " + str(data[2]))

print("\tQuantity: " + str(data[3]))

str(input("\tEnter Any Key To Continue \n\n"))

var = True

else:

var = False

if (var == False):

print("\tNo Medicine Found")

def ismedalreadytaken(self):

self.medicines = []

self.medicinedetails()

if (binary\_search(self.medicines, 0, len(self.medicines) - 1, self.medicineid) == -1):

return False

else:

return True

def AddMedicine(self):

if (self.ismedalreadytaken() == False):

if (int(self.quantity) > 0):

f = open("medicines.txt", "a+")

f.write(str(self.medicineid) + ","

+ str(self.medicinename) + ","

+ str(self.price) + ","

+ str(self.quantity) + ","

+ "\n")

print('\n\tMedicine Successfully Added')

else:

print("\tEnter Quantity greater than zero")

else:

print("\tMedicine ID is Already Taken")

str(input("\tEnter any Key To Continue"))

def EditMedicine(self):

file = "medicines.txt"

file2 = "tempmedicines.txt"

any\_var = open(file2, "w")

for line in open(file, "r").readlines():

data = line.split(',')

if (str(self.medicineid) == str(data[0])):

any\_var.write(str(input('\tEnter New ID: ')) + ","

+ 'none' + ","

+ str(input('\tEnter New Price: ')) + ","

+ str(input('\tEnter Your New Quantity: '))

+ ",\n")

else:

any\_var.write(str(data[0]) + "," +

str(data[1]) + "," +

str(data[2]) + "," +

str(data[3]) + ",\n")

any\_var.close()

str(input("\n\tSuccessfully Edited\n\tPress Any Key To Go Back To The main menu\n\n"))

self.FileDeletion(file2, file)

def DeleteMedicine(self):

file = "medicines.txt"

file2 = "tempmedicines.txt"

First\_file = open(file, "r")

Second\_file = open(file2, "w")

for line in First\_file.readlines():

data = line.split(',')

# Linear Search

if (str(self.medicineid) == str(data[0])):

continue

else:

Second\_file.write(

str(data[0]) + "," +

str(data[1]) + "," +

str(data[2]) + "," +

str(data[3]) + ",\n")

First\_file.close()

Second\_file.close()

str(input("\n\tSuccessfully Deleted\n\tGo Back to Menu\n\n"))

self.FileDeletion(file2, file)

def DeleteAllMedicines(self):

asking = input("Do You Want To Delete All Data????? If Yes Than press Y else any key to cancel")

if (os.path.exists("medicines.txt")) and (asking == 'Y'):

os.remove("medicines.txt")

class Admin:

users = []

def \_\_init\_\_(self, name, email, password, a):

self.name = name.lower()

self.email = email

self.password = password

self.a = a

def getuserids(self):

file = "Admins.txt"

if os.path.exists(file):

for line in open(file, "r").readlines():

data = line.split(',')

self.users.append(data[1])

def isUserAlreadyRegistered(self):

self.users =[]

self.getuserids()

if(linearsearch(self.users,self.email)==-1):

return False

else:

return True

def Register(self):

if (True):

f = open("Admins.txt", "a+")

f.write(str(self.name) + "," + str(self.email) + ","+ str(self.password)+ ","+ str(self.a)+"\n")

print("\tSuccessFully Registered")

else:

print("\tEmail already Taken")

str(input("Press any Key To Move Further"))

def Login(self):

any\_var = ''

if (os.path.exists("Admins.txt")):

for line in open("Admins.txt", "r").readlines():

data = line.split(',')

if (self.email == "" and self.password == ""):

print("Please Type Anything:")

elif (self.email == data[1] and self.password == data[2]):

self.name = data[0]

self.email = data[1]

self.password = data[2]

self.MainMenu()

else:

x = 'notfound'

if (any\_var == 'notfound'):

print("\tInvalid Email or Password")

else:

print("\n\tNo User Registered")

def MainMenu(self):

while (True):

print("\t\t\t " + self.name + "! Select anything from ZAMA")

print("-------------------------------------------------------------------------------------------------------------")

print(

"\tPurchase Medicine \tAdd Medicine \tEdit Medicine \tSearch Medicine \tDelete Medicine \n\tDelete All Medicines \tDisplay Medicines \t Log Out")

print()

print(

"-------------------------------------------------------------------------------------------------------------")

choice = input("\n\tChoose your option:\n Press 1 to Purchase\n Press 2 to Add\n: Press 3 to Edit\n Press 4 to Search\n Press 5 to Delete\n Press 6 to Delete ALl\n Press 7 to DisplayAll\n Press 8 to Available \n Press 9 to Log out\n\n")

print("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_")

if choice == '1':

print("============================================")

medicines = Medicines('none', 'none', 'none', 'none')

medicines.PurchaseMedicine()

print("============================================")

elif choice == '2':

medicine = Medicines(input("\t|| Enter Medicine ID: ||"),

input("\t|| Enter Medicine Name: ||"),

input("\t|| Enter Medicine Price: ||"),

input("\t|| Enter Medicine Quantity: ||"),

)

medicine.AddMedicine()

print()

print("===================================================")

elif choice == '3':

medicines = Medicines(input("\t||Enter Medicine ID: ||"), '|| none ||', '|| none ||', '|| none ||')

medicines.EditMedicine()

print()

print("===================================================")

elif choice == '4':

medicines = Medicines(input("\t|| Enter Medicine ID: ||"), '|| none ||','|| none ||', '|| none ||')

medicines.SearchMedicine()

print()

print("===================================================")

elif choice == '5':

medicines = Medicines(input("\tEnter Medicine ID:"), 'none', 'none', 'none')

medicines.DeleteMedicine()

print()

print("===================================================")

elif choice == '6':

medicines = Medicines('none', 'none', 'none', 'none')

medicines.DeleteAllMedicines()

print()

print("===================================================")

elif choice == '7':

medicines = Medicines('none', 'none', 'none', 'none')

medicines.DisplayAllMedicine()

print()

print("===================================================")

elif choice == '8':

medicines = Medicines('none', 'none', 'none', 'none')

medicines.DisplayAvailableMedicines()

print()

print("===================================================")

elif choice == '9':

break

else:

print("\tInvalid Input")

def MainMethod():

t1 = threading.Thread(target=Method1(), args=(10,))

t2 = threading.Thread(target=SortingMethod(), args=(10,))

# starting thread 1

t1.start()

# starting thread 2

t2.start()

# wait until thread 1 is completely executed

t1.join()

# wait until thread 2 is completely executed

t2.join()

def Method1():

while (True):

print('\t\t\t\t-----------------------------------------------------')

print('\t\t\t\t ZAMA PHARMACY MANAGEMENT SYSTEM ')

print('\t\t\t\t-----------------------------------------------------')

print()

print("Note! if you already have an account than press 1 to Login or if You want to make account press 2: ")

#press 3 To exit

print()

print("===================================================")

print()

print("\tLog\_in \t\t\t\t\t\tRegister\t\t\t\t\t\tEnd")

print()

choice = input("Choose Your Option: ")

if choice == '1':

admin = Admin('none', input("\tEnter Your Email:"), input("\tEnter Your Password:"), 'none')

admin.Login()

elif choice == '2':

admin = Admin(

input("\tEnter Your Name:"), input("\tEnter Your Email:"), input("\tEnter Your Password:"),int(input(" Enter Your Balance ")))

admin.Register()

elif choice == '3':

break

else:

print('Wrong Input')

print()

print("===================================================")

def SortingMethod():

ask = input("Press Y to Sort Medicine File")

if(ask =='Y'):

inputFile = open("medicines.txt", 'r')

lineList = inputFile.readlines()

lineList.sort()

print(lineList)

os.remove("sorted\_medicines\_files.txt")

for line in lineList:

print(line)

with open('sorted\_medicines\_files.txt', 'a') as f:

lineList.sort()

f.write(line)

ask = input("Press Y to Sort Admin File")

if(ask =='Y'):

inputFile = open("Admins.txt", 'r')

lineList = inputFile.readlines()

lineList.sort()

os.remove("sorted\_Admins.txt")

print(lineList)

for line in lineList:

print(line)

with open('sorted\_Admins.txt', 'a') as f:

lineList.sort()

f.write(line)

MainMethod()