Dr. D. Y. PATIL VIDYAPEETH, PUNE
(Deemed to be University)

DR. D. Y. PATIL VIDYAPEETH

PIMPRI, PUNE – 411 018

DR. D. Y. PATIL SCHOOL OF SCIENCE & TECHNOLOGY

TATHAWADE, PUNEs

A Project Based Learning Report on

PHARMACY MANAGEMENT SYSTEM

Submitted By

NAME OF MEMBERS	ROLL NUMBER
1. ADITI BANE	05
2. VAISHALI KHANDAR	26
3. SHRAVANI JADHAV	21

ARTIFICIAL INTELLIGENCE & DATASCIENCE

ACADEMIC YEAR 2022-2023

Dr. D. Y. PATIL VIDYAPEETH, PUNE
(Deemed to be University)

DR. D. Y. PATIL VIDYAPEETH

PIMPRI, PUNE – 411 018

DR. D. Y. PATIL SCHOOL OF SCIENCE AND TECHNOLOGY

TATHAWADE, PUNEs

CERTIFICATE

This is to certify that the Project Report entitled

PHARMACY MANAGEMENT SYSTEM

Is a bonafide work carried out by the students under the supervision of **Mrs. Akanksha Goel** and it is submitted towards the partial fulfillment of the requirement Project Based Learning

Mrs. Akanksha Goel — Subject Teacher

Prof J.K Pal — Director

ARTIFICIAL INTELLIGENCE & DATASCIENCE

ABSTRACT

This Pharmacy Management System project's goal is to enhance the management and use of medications in healthcare. The Pharmacy Management System is a tool that pharmacists can use to methodically manage their businesses. The Pharmacy Management System can assist by presenting information on the medication when the name of the medication is input. The dosage and expiration date of the medication are displayed on a computer. It becomes quite difficult to manually handle all the drug specifications in huge medical supply stores. Using this pharmacy management system, we can keep track of all the medications.

INDEX

➢ PROBLEM STATEMENT

To Design Software That Will Manage a Pharmacy's Database.

➢ INTRODUCTION

A pharmacy management system is a type of management that is intended to boost efficiency, safety, and accuracy in a pharmacy. Any pharmacy that needs to keep a database can use this application. It is a computer-based system that aids pharmacists in managing inventories, costs, and other factors including medical safety. Depending on the user's needs, the software can provide reports. During opening stock and sales transactions, the system allows the user to enter the manufacturing and expiry dates for a specific product or medication. The system will also provide a report with a list of products that will expire after a certain date before the product itself does. Additionally, it requires manual entry when fresh batches of medications arrive and when they are taken temporarily out of the pharmacy. Every month, the pharmacist might want to create a report for the flow of pharmaceuticals into and out of the pharmacy. This report would include information about the drugs, such as their expiration date, date of purchase, remaining supply, and location within the drugstore. Given that Ethiopia is moving toward pharmaceutical patient care, this will increase the effectiveness of clinical work and make it more convenient for patients. Additionally, including secure medication data storage, quick medication search, delete, and update functions.

➢ OBJECTIVE

The Pharmacy Management System's primary goal is to handle the specifics of the medications, stocks, inventory, pharmacies, and sales. It oversees the management of all data relating to drugs, companies, and drug sales. The project's goal is to create a software application that will reduce the amount of manual effort required to manage the company's inventory, stocks, and medicines. It keeps track of every aspect of the inventory, pharmacy, and sales.

➢ SCOPE

The project's objectives are restricted to those of a drugstore, which include enhancing access to care in the Estate and neighboring communities, lowering the number of hospital and long-term care admissions, and making the greatest use of available resources. It is necessary and a crucial component of any contemporary culture that is always changing to utilize a computer-based management system to increase a pharmacy's efficiency. Drug prescriptions and drug interactions won't be handled by the system. Contraindication and polypharmacy will need to be manually handled by the pharmacist because the system will not be able to manage them in a prescription.

➢ Overview

The pharmacy management system provides functions such as identifying medication usage instructions, minimizing human errors in medication safety, facilitating access to drug information and information management among employees, providing optimal drug movement in the pharmacy unit, and enabling report generation in a significantly short period of time, despite the use of a database for the purpose. The system will solve the current system's problem by minimizing time waste and reducing resources by simply switching from manual to computerized systems.

➢ FUNCTIONAL REQUIREMEMTS

The system performs functions such as storing necessary drug information, easily searching for medicine, updating, deleting, and saving medicine data

- ✓ Generate report: The pharmacy management system generates a week report on drug information and exports it as an output document.
- ✓ Store the necessary information of the drugs: The pharmacy system stores detailed information about each medicine, such as its actual name, formula, how important it is, and for which diseases it is required. because information about each drug was required in some cases, such as how the drug was used, when it was used, and for whom it was given.
- ✓ Searching medicine and other data's: The pharmacy system allows for easy searching of medicines based on where they are stored and how they behave. The searching procedure is based on the name of the given data or the identification of the item. Here, when the user searches the item on the search bar, related itemsare displayed on the screen and the user can select the actual item that the user requires.
- ✓ Altering pharmacy data in the system: Changing medicines to another because one is outdated, modifying saved medicine data for incorrect data, and deleting pharmacy data can all be done on the system.

➢ NON-FUNCTIONAL REQUIREMENT

This pharmacy management system is capable of the following functions.

- ✓ Usability: Because the system has a user-friendly user interface, anyone who is familiar with Windows operations can operate it. which has instruction menus on how to use it, and which self-directed application can then be used by the system without ambiguity.
- ✓ Reliability: The pharmacy system is available based on user requirements, can function properly, and perform transactions efficiently, including safe pharmacy data management. In the event of an invalid or malfunctioning operation, the system will restart within 5 seconds to prevent data loss and ensure safe operation. To change anything on the pharmacy system, you must enter a password. The

pharmacist manager has control over the system by logging into the pharmacy system. Any user cannot use the system, but a guest user can view general properties of the pharmacy and medicines without a password; as a result, data is protected and controlled by only the administrator.

- ✓ Performance: The Pharmacy Management System Performs Its Function in Less Than Two Seconds and Can Be Accessed by One User at A Time or Concurrently. The User Must First Log in To the System, Which Must Have the Pharmacy System Privileged and Can Also Store Data Up To 16 Gb, In Order to Access. The System May Wait Up to One Minute Before Restarting If It Is Busy Due to Malfunction Operation; Otherwise, The Pharmacy System Restarts.
- ✓ Operation: For safe work, the pharmacy manager operates and controls the pharmacy management system.
- ✓ Supportability: The pharmacy management system is compatible with any version of the Windows operating system, including Windows XP, Windows 7, Windows 10, and others. The manager of the pharmacy system documents the system for easy maintenance, so the system can be easily maintained. System developers maintain it in other ways for corrective and other severe problems.

➢ MODULE

1. Login: This module will grant the pharmacist system access. They must enter the user ID and password that were assigned to them during registration.
2. Register: The pharmacist will need to enter some basic information about themselves into the system before receiving an ID and password to access the system.
3. Record: This module will include a function for storing drug information. While it will provide information on dosage, cost, and storage location.
4. Notification: This module will notify the pharmacist when the drugs are running out.

➢ ALGORITHM

1. Import Modules
2. Initializing the window
3. Function to add items
4. Function to delete items
5. Function for making list
6. Function to Update items
7. Function to search items
8. Function to clear screen
9. Exit function
10. Define buttons labels and entry widget

➢ FUNCTIONS USED IN CODE

- Tkinter module – Tkinter module is used for creating Graphical user interface in python.
- tkinter import * – import everything from the module.
- from tkinter import messagebox – Import message box separately for showing messages on the screen.
- import os – This is the functional module in order to get the data from your computer.
- f – Variable for storing project database file
- open – Before doing any operation on a file we have to open the file. Open function opens a particular file.
- root – Initializing the root window of Pharmacy management system.
- root.title – Use to set title to window.
- root.configure – Used to configure attributes to the window, such as width, height, background colour.
- Add_Items() – Function for adding new items.
- Declaring global function var

- r – Opens a file in read mode.
- .get() – It returns the value of the name and number with the specific key.
- .delete() – To remove a specific entry.
- After successful addition of a new item message will pop up on the screen that “Item added successfully!!”
- Delete_Items() – Function for deleting items.
- w – Opens a file in write mode.
- . write () – This method is used to write text in the file.
- os.remove() – To remove or delete file path
- os.rename() – Used to rename a file or directory.
- f.close() – Used to closes opened file
- After successful addition of a new item message will pop up on the screen that “Item deleted successfully!!”
- list() – Function for making a list of items.
- f.seek() – seek() function is used to change the position of the File Handle to a given specific position.
- .insert()- Insert the elements in the list.
- Search_Item() – Function to search items.
- open – Before doing any operation on a file we have to open the file. Open function opens a particular file.
- If item is present in the list message box will show “Error end of file” and if its not present in the list message box will show “NOT FOUND”
- .close() – Used to close an opened file.
- Clear_Item() – Function for clearing screen.

Source Code

```
from tkinter import *
from tkinter import messagebox
import os
f=open("database_proj",'a+')
root = Tk()
root.title("Simple Pharmacy Managment System")
root.configure(width=1500,height=600,bg='BLACK')
var=-1


def additem():
    global var
    num_lines = 0
    with open("database_proj", 'r') as f10:
        for line in f10:
            num_lines += 1
    var=num_lines-1
    e1= entry1.get()
    e2=entry2.get()
    e3=entry3.get()
    e4=entry4.get()
    e5=entry5.get()
    f.write('{0} {1} {2} {3} {4}\n'.format(str(e1),e2,e3,str(e4),e5))
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)


def deleteitem():
    e1=entry1.get()
```

```
    with open(r"database_proj") as f, open(r"database_proj1", "w") as working:
        for line in f:
            if str(e1) not in line:
                working.write(line)
    os.remove(r"database_proj")
    os.rename(r"database_proj1", r"database_proj")
    f.close()
    working.close()
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)

def firstitem():
    global  var
    var=0
    f.seek(var)
    c=f.readline()
    v=list(c.split(" "))
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)
    entry1.insert(0,str(v[0]))
    entry2.insert(0,str(v[1]))
    entry3.insert(0,str(v[2]))
    entry4.insert(0,str(v[3]))
    entry5.insert(0,str(v[4]))
```

```
def nextitem():
    global var
    var = var + 1
    f.seek(var)
    try:
        c=f.readlines()
        xyz = c[var]
        v = list(xyz.split(" "))
        entry1.delete(0, END)
        entry2.delete(0, END)
        entry3.delete(0, END)
        entry4.delete(0, END)
        entry5.delete(0, END)
        entry1.insert(0, str(v[0]))
        entry2.insert(0, str(v[1]))
        entry3.insert(0, str(v[2]))
        entry4.insert(0, str(v[3]))
        entry5.insert(0, str(v[4]))
    except:
        messagebox.showinfo("Title", "SORRY!...NO MORE RECORDS")

def previousitem():
        global var
        var=var-1
        f.seek(var)
        try:
            z = f.readlines()
            xyz=z[var]
            v = list(xyz.split(" "))
            entry1.delete(0, END)
            entry2.delete(0, END)
```

```
            entry3.delete(0, END)
            entry4.delete(0, END)
            entry5.delete(0, END)

            entry1.insert(0, str(v[0]))
            entry2.insert(0, str(v[1]))
            entry3.insert(0, str(v[2]))
            entry4.insert(0, str(v[3]))
            entry5.insert(0, str(v[4]))
        except:
            messagebox.showinfo("Title", "SORRY!...NO MORE RECORDS")

def lastitem():
    global var
    f4=open("database_proj",'r')
    x=f4.read().splitlines()
    last_line= x[-1]
    num_lines = 0
    with open("database_proj", 'r') as f8:
        for line in f8:
            num_lines += 1
    var=num_lines-1
    print(last_line)
    try:
        v = list(last_line.split(" "))
        entry1.delete(0, END)
        entry2.delete(0, END)
        entry3.delete(0, END)
        entry4.delete(0, END)
        entry5.delete(0, END)
```

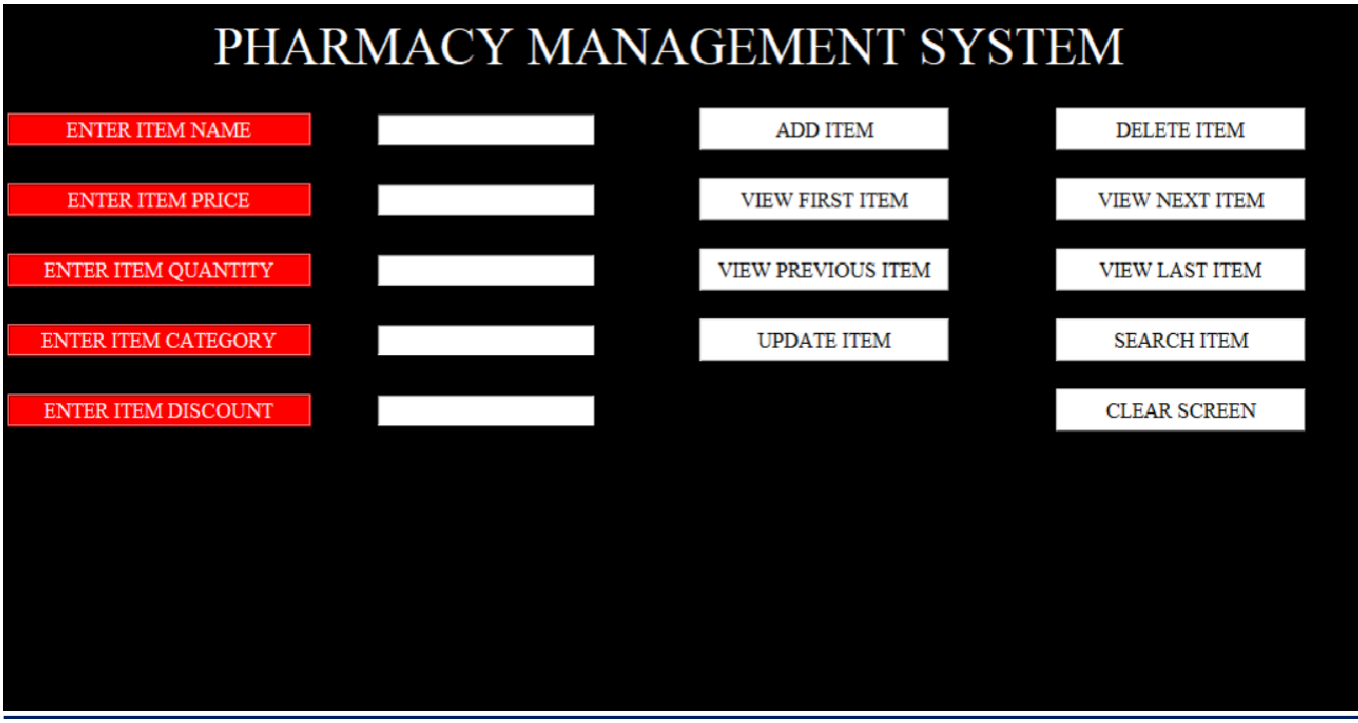
```
            entry1.insert(0, str(v[0]))
            entry2.insert(0, str(v[1]))
            entry3.insert(0, str(v[2]))
            entry4.insert(0, str(v[3]))
            entry5.insert(0, str(v[4]))
        except:
            messagebox.showinfo("Title", "SORRY!...NO MORE RECORDS")

def updateitem():
    e1 = entry1.get()
    e2 = entry2.get()
    e3 = entry3.get()
    e4 = entry4.get()
    e5 = entry5.get()
    with open(r"database_proj") as f1, open(r"database_proj1", "w") as working:
        for line in f1:
            if str(e1) not in line:
                working.write(line)
            else:
                working.write('{0} {1} {2} {3} {4}'.format(str(e1), e2, e3, str(e4), e5))
    os.remove(r"database_proj")
    os.rename(r"database_proj1", r"database_proj")

def searchitem():
    i=0
    e11 = entry1.get()
    with open(r"database_proj") as working:
        for line in working:
            i=i+1
            if str(e11) in line:
                break
```

```
        try:
            v = list(line.split(" "))
            entry1.delete(0, END)
            entry2.delete(0, END)
            entry3.delete(0, END)
            entry4.delete(0, END)
            entry5.delete(0, END)
            entry1.insert(0, str(v[0]))
            entry2.insert(0, str(v[1]))
            entry3.insert(0, str(v[2]))
            entry4.insert(0, str(v[3]))
            entry5.insert(0, str(v[4]))
        except:
            messagebox.showinfo("Title", "error end of file")
    working.close()

def clearitem():
    entry1.delete(0, END)
    entry2.delete(0, END)
    entry3.delete(0, END)
    entry4.delete(0, END)
    entry5.delete(0, END)

label0=Label(root,text="PHARMACY MANAGEMENT SYSTEM
",bg="black",fg="white",font=("Times", 30))
label1=Label(root,text="ENTER ITEM
NAME",bg="red",relief="ridge",fg="white",font=("Times", 12),width=25)
entry1=Entry(root , font=("Times", 12))
label2=Label(root, text="ENTER ITEM
PRICE",bd="2",relief="ridge",height="1",bg="red",fg="white", font=("Times",
12),width=25)
```

```
entry2= Entry(root, font=("Times", 12))
label3=Label(root, text="ENTER ITEM
QUANTITY",bd="2",relief="ridge",bg="red",fg="white", font=("Times",
12),width=25)
entry3= Entry(root, font=("Times", 12))
label4=Label(root, text="ENTER ITEM
CATEGORY",bd="2",relief="ridge",bg="red",fg="white", font=("Times",
12),width=25)
entry4= Entry(root, font=("Times", 12))
label5=Label(root, text="ENTER ITEM
DISCOUNT",bg="red",relief="ridge",fg="white", font=("Times", 12),width=25)
entry5= Entry(root, font=("Times", 12))
button1= Button(root, text="ADD ITEM", bg="white", fg="black", width=20,
font=("Times", 12),command=additem)
button2= Button(root, text="DELETE ITEM", bg="white", fg="black", width =20,
font=("Times", 12),command=deleteitem)
button3= Button(root, text="VIEW FIRST ITEM" , bg="white", fg="black", width
=20, font=("Times", 12),command=firstitem)
button4= Button(root, text="VIEW NEXT ITEM" , bg="white", fg="black", width
=20, font=("Times", 12), command=nextitem)
button5= Button(root, text="VIEW PREVIOUS ITEM", bg="white", fg="black",
width =20, font=("Times", 12),command=previousitem)
button6= Button(root, text="VIEW LAST ITEM", bg="white", fg="black", width
=20, font=("Times", 12),command=lastitem)
button7= Button(root, text="UPDATE ITEM", bg="white", fg="black", width =20,
font=("Times", 12),command=updateitem)
button8= Button(root, text="SEARCH ITEM", bg="white", fg="black", width =20,
font=("Times", 12),command=searchitem)
button9= Button(root, text="CLEAR SCREEN", bg="white", fg="black", width=20,
font=("Times", 12),command=clearitem)
label0.grid(columnspan=6, padx=10, pady=10)
```

```
label1.grid(row=1,column=0, sticky=W, padx=10, pady=10)
label2.grid(row=2,column=0, sticky=W, padx=10, pady=10)
label3.grid(row=3,column=0, sticky=W, padx=10, pady=10)
label4.grid(row=4,column=0, sticky=W, padx=10, pady=10)
label5.grid(row=5,column=0, sticky=W, padx=10, pady=10)
entry1.grid(row=1,column=1, padx=40, pady=10)
entry2.grid(row=2,column=1, padx=10, pady=10)
entry3.grid(row=3,column=1, padx=10, pady=10)
entry4.grid(row=4,column=1, padx=10, pady=10)
entry5.grid(row=5,column=1, padx=10, pady=10)
button1.grid(row=1,column=4, padx=40, pady=10)
button2.grid(row=1,column=5, padx=40, pady=10)
button3.grid(row=2,column=4, padx=40, pady=10)
button4.grid(row=2,column=5, padx=40, pady=10)
button5.grid(row=3,column=4, padx=40, pady=10)
button6.grid(row=3,column=5, padx=40, pady=10)
button7.grid(row=4,column=4, padx=40, pady=10)
button8.grid(row=4,column=5, padx=40, pady=10)
button9.grid(row=5,column=5, padx=40, pady=10)
root.mainloop()
```

➢ OUTPUT



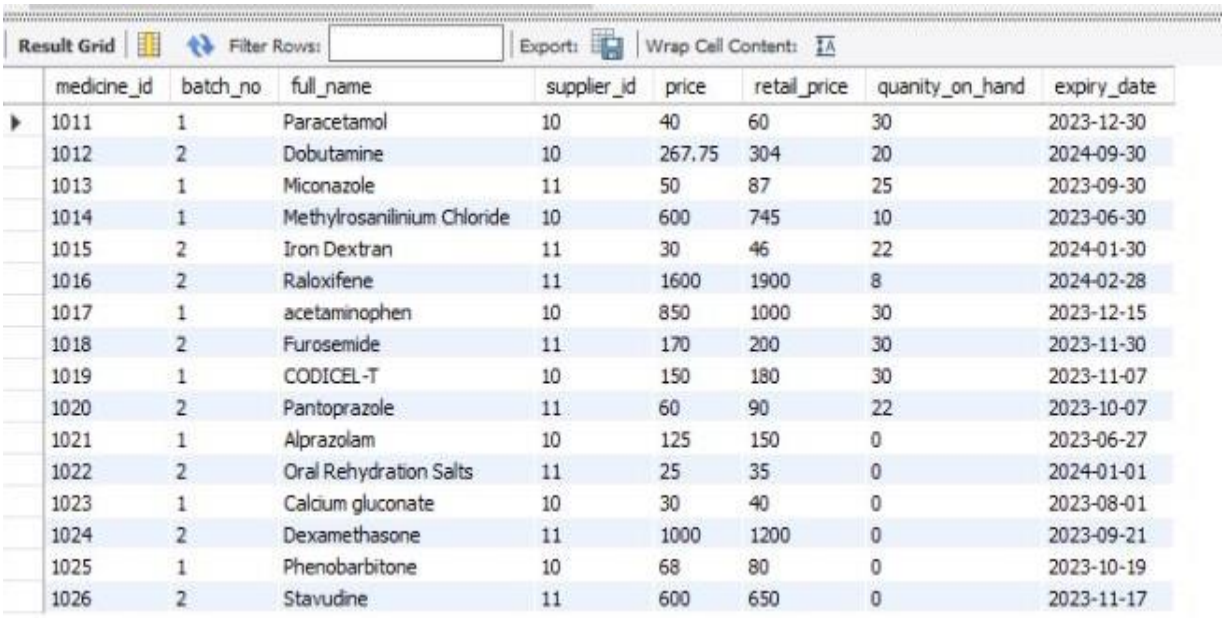
➢ DATABASE

```
create database pharmacy;
use pharmacy;

CREATE TABLE user (
user_id int(11) NOT NULL,
username varchar(30) NOT NULL,
password varchar(20) NOT NULL,
fullname varchar(100) NOT NULL,
user_type int(1) NOT NULL);

Insert into user values
('11','owner_1','124578','Jayesh Gohil');
('12','assistant_1','987654','Nikhil Shinde');
('13','manager','567932','SnehaDeshmukh')
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

user_id	username	password	fullname
11	owner_1	124578	Jayesh Gohil
12	assistant_1	987654	Nikhil Shinde
13	manager	567932	Sneha Deshmukh

```
CREATE TABLE medicine (
medicine_id int(11) NOT NULL,
batch_no varchar(15) NOT NULL,
full_name varchar(50) NOT NULL,
supplier_id int(11) NOT NULL,
price float NOT NULL,
retail_price float NOT NULL,
quanity_on_hand int(6) NOT NULL,
expiry_date date NOT NULL);

Insert into medicine values
('1011','1','Paracetamol','10','40','60','30','2023-12-30');
('1012','2','Dobutamine','10','267.75','304','20','2024-9-30');
('1013','1','Miconazole','11','50','87','25','2023-9-30');
('1014','1','Methylrosanilinium Chloride','10','600','745','10','2023-6-30');
('1015','2','Iron Dextran','11','30','46','22','2024-1-30');
('1016','2','Raloxifene','11','1600','1900','8','2024-02-28');
('1017','1','acetaminophen','10','850','1000','30','2023-12-15');
('1018','2','Furosemide','11','170','200','30','2023-11-30');
('1019','1','CODICEL-T','10','150','180','30','2023-11-07');
('1020','2','Pantoprazole','11','60','90','22','2023-10-07');
('1021','1','Alprazolam','10','125','150','0','2023-06-27');
('1022','2','Oral Rehydration Salts','11','25','35','0','2024-01-01');
('1023','1','Calcium gluconate','10','30','40','0','2023-08-01');
('1024','2','Dexamethasone','11','1000','1200','0','2023-09-21');
('1025','1','Phenobarbitone','10','68','80','0','2023-10-19');
('1026','2','Stavudine','11','600','650','0','2023-11-17');
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

medicine_id	batch_no	full_name	supplier_id	price	retail_price	quanity_on_hand	expiry_date
1011	1	Paracetamol	10	40	60	30	2023-12-30
1012	2	Dobutamine	10	267.75	304	20	2024-09-30
1013	1	Miconazole	11	50	87	25	2023-09-30
1014	1	Methylrosanilinium Chloride	10	600	745	10	2023-06-30
1015	2	Iron Dextran	11	30	46	22	2024-01-30
1016	2	Raloxifene	11	1600	1900	8	2024-02-28
1017	1	acetaminophen	10	850	1000	30	2023-12-15
1018	2	Furosemide	11	170	200	30	2023-11-30
1019	1	CODICEL-T	10	150	180	30	2023-11-07
1020	2	Pantoprazole	11	60	90	22	2023-10-07
1021	1	Alprazolam	10	125	150	0	2023-06-27
1022	2	Oral Rehydration Salts	11	25	35	0	2024-01-01
1023	1	Calcium gluconate	10	30	40	0	2023-08-01
1024	2	Dexamethasone	11	1000	1200	0	2023-09-21
1025	1	Phenobarbitone	10	68	80	0	2023-10-19
1026	2	Stavudine	11	600	650	0	2023-11-17

CREATE TABLE supplier (
supplier_id int(11) NOT NULL,
supplier_name varchar(50) NOT NULL,
supplier_mobile varchar(20) NOT NULL,
status int (1) NOT NULL);

Insert into supplier values
('10','Fresh Point Supply','9897652341','1');
('11','Highland Supply','7768954321','1');

supplier_id	supplier_name	supplier_mobile	status
10	Fresh Point Supply	9897652341	1
11	Highland Supply	7768954321	1

CREATE TABLE received (
receiving_id int(11) NOT NULL,
batch_no varchar (30) NOT NULL,
supplier_id int(11) NOT NULL,
medicine_id int(11) NOT NULL,
price float NOT NULL,
quantity int(6) NOT NULL,
amount float NOT NULL,
date_received date NOT NULL);

Insert into received values
('20','1','10','1014','600','5','5','2022-12-29');
('21','2','11','1016','1600','7','14','2023-01-03');
('22','2','11','1015','30','8','8','2022-12-29');
('23','1','11','1013','50','5','5','2022-12-29');
('24','2','11','1020','60','8','8','2023-01-03');

receiving_id	batch_no	supplier_id	medicine_id	price	quantity	amount	date_received
20	1	10	1014	600	5	5	2022-12-29
21	2	11	1016	1600	7	14	2023-01-03
22	2	11	1015	30	8	8	2022-12-29
23	1	11	1013	50	5	5	2022-12-29
24	2	11	1020	60	8	8	2023-01-03

CREATE TABLE returned (
return_id int(11) NOT NULL,
batch_no varchar(15) NOT NULL,
supplier_id int(11) NOT NULL,
medicine_id int(11) NOT NULL,

price float NOT NULL,
quantity int (6) NOT NULL,
amount float NOT NULL,
date_returned date NOT NULL);

Insert into returned values
('31','1','10','1011','40','10','20','2023-01-10');
('32','2','11','1018','170','8','8','2023-01-09');
('33','2','10','1012','170','5','5','2023-01-10');
('34','1','10','1017','850','10','10','2023-01-10');
('35','1','10','1019','150','7','14','2023-01-10');

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

return_id	batch_no	supplier_id	medicine_id	price	quantity	amount	date_returned
31	1	10	1011	40	10	20	2023-01-10
32	2	11	1018	170	8	8	2023-01-09
33	2	10	1012	170	5	5	2023-01-10
34	1	10	1017	850	10	10	2023-01-10
35	1	10	1019	150	7	14	2023-01-10

CREATE TABLE stock_out (
stock_out_id int (11) NOT NULL,
batch_no varchar (15) NOT NULL,
supplier_id int (11) NOT NULL,
medicine_id int(11) NOT NULL,
price float NOT NULL,
quantity int(6) NOT NULL,
amount float NOT NULL);

Insert into stock_out values
('101','1','10','1021','125','10','10');
('102','2','11','1022','125','25','25');
('103','1','10','1023','30','15','30');
('104','2','11','1024','1000','7','35');
('105','1','10','1025','68','20','40');

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

stock_out_id	batch_no	supplier_id	medicine_id	price	quantity	amount
101	1	10	1021	125	10	10
102	2	11	1022	125	25	25
103	1	10	1023	30	15	30
104	2	11	1024	1000	7	35
105	1	10	1025	68	20	40
106	2	11	1026	600	10	10

Conclusion

Today, management is one of the most important aspects of any form. Management provides the sophistication to perform any type of task in a specific format. The pharmacy management system is used to manage the majority of pharmacy-related activities in the pharmacy. The primary goal is to improve accuracy while also improving safety and efficiency in the pharmaceutical industry. We can also include a bar-code facility in this project by using a bar-code reader, which will detect the expiry date and other relevant information about the medicines. We could also conclude that by utilizing pharmacy software, processing new prescriptions and refills is quick and easy with the new, simple to learn and use Graphical User Interface (GUI) pharmacy management solution, which requires only a few keystrokes or mouse clicks. That is due to automation, which allows the pharmacist to complete his or her work much more quickly, as well as a shift from product-oriented to patient-oriented care, which is one of the most crucial aspects of pharmaceutical care. In other words, the pharmacist will have more time to spend counselling his or her customers, with the goal of patient counselling being one of the most important ways to avoid medication errors.