Drug inventory management system

1. Introduction

Managing pharmaceutical stock is a critical aspect of health service delivery, especially in hospital clinics and pharmacies, inadequate drug inventory control can lead to shortages, overstocking or the us f expired medicines al o which ending patients health and increase operational costs [1],[5]. The project proposes the development of a drug inventory management system designed to automate and properly streamline the process of managing drug inventory in health facility

2. Problem statement

Many clinics and pharmacies in a a resource limited setting rely on a manual or semi - automated methods for inventory tracking and management. Hese methods are prone to man error, and real-time updates, and are inefficient in alerting the responsible about critical issues like stock depletion expiration [2],[3]. The proposed system aims to resolve these challenges by offering a digital solution that improves efficiency, reliability and safety.

3. Objectives

- To develop a web-based application for managing drug inventory in a health
- Facility
- To implement features for tacking stock levels , expiry dates and generating
- alerts for expiring drugs and low tock.
- To enhance data accuracy and accessibility through a user- friendly interface
- To ensure clear demonstration of OOP principles ie encapsulation, inheritance, abstraction and polymorphism

4. Methodology

The system was developed using an agile software development approach to enable iterative progress. Continuous feedback and flexibility in addressing user requirements .

The key technologies and tools used include:

Frontend: HTML and CSS

Backend: python with flask

5. System Architecture

Componens within the system

- app.py :this acts as the main application controller
- models/ : this houses classes for drugs, users and inventory
- templates/ : this holds the html templates for the user interface

6. **implementation**

front end: this has the login form simulating user access, the dashboard to view drug inventory and also forms for adding drugs, stocking in and viewing low stock drugs

backend: this handles drug registration and management, stock in/out operations, expiry date validation low stock filtering and user role simulation using OOP

Key classes:

- **Drug:** attributes include name, type, batch number, expiry date and quantity. This demonstrates encapsulation and abstraction
- **User and pharmacist** this demonstrates inheritance and polymorphism through methods like get_role()
- **Inventory**: this manages drug in memory; supports add, retrieve and list operations

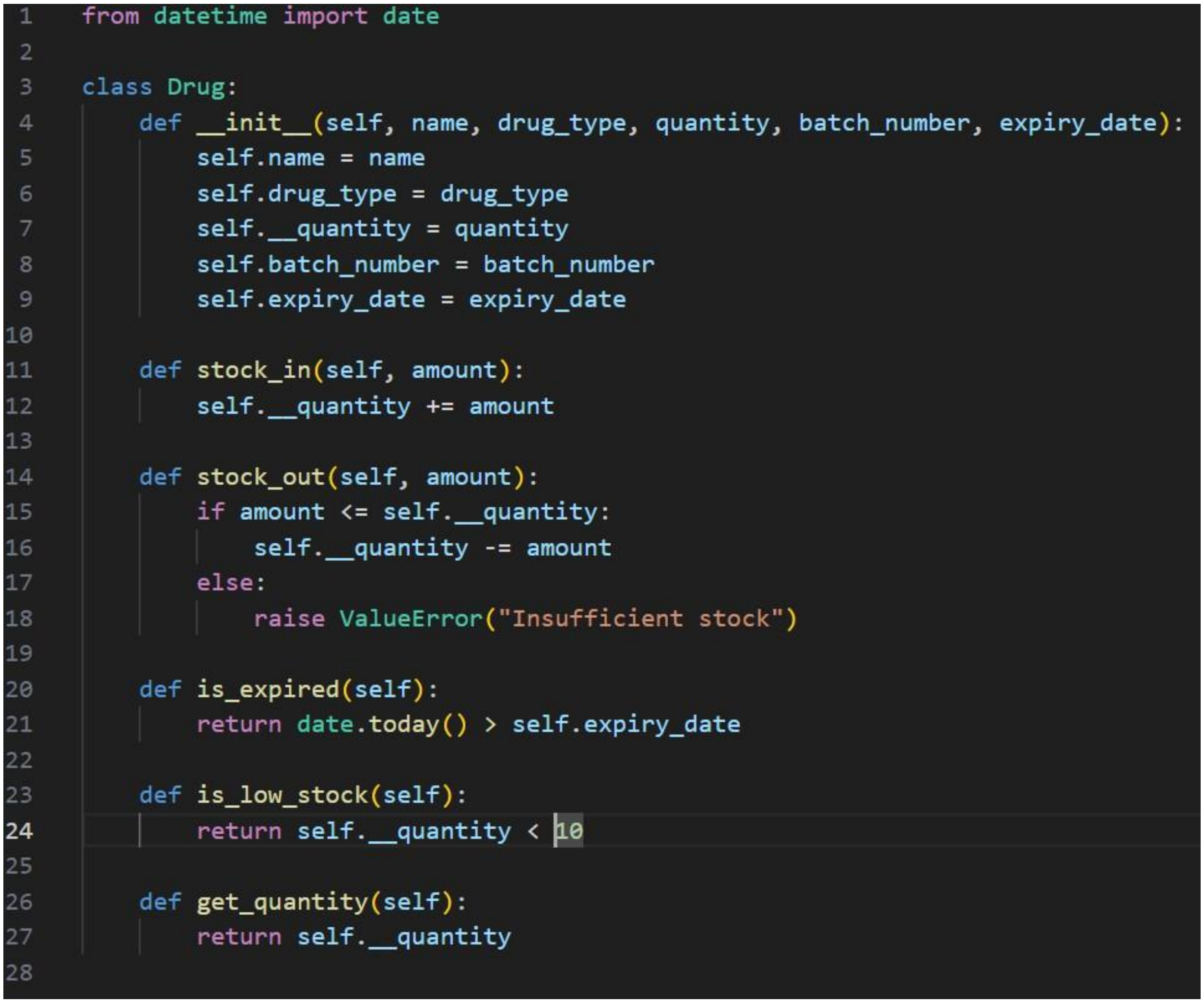
Results

The system allows

- Drug registration and inventory updates
- Expiry and low stock alerts
- Simulation of hamacist roles
- Efficient use of op cncept and olar python code

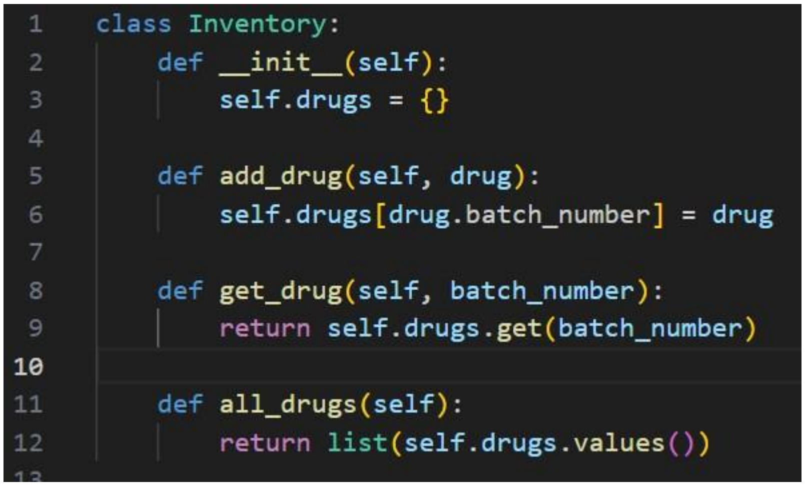
Back end implementation was done using flasks shown below

Drug.py



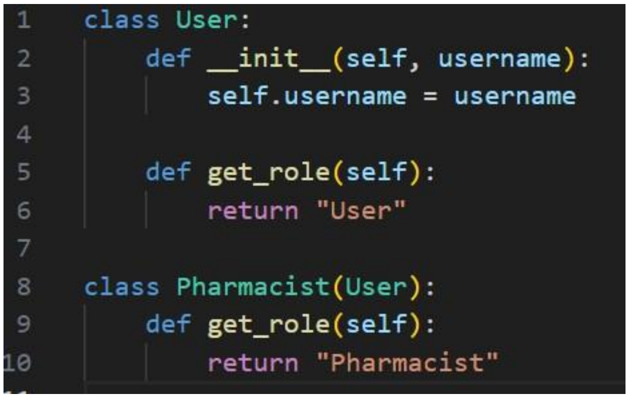
```python
from datetime import date

class Drug:
    def __init__(self, name, drug_type, quantity, batch_number, expiry_date):
        self.name = name
        self.drug_type = drug_type
        self.__quantity = quantity
        self.batch_number = batch_number
        self.expiry_date = expiry_date

    def stock_in(self, amount):
        self.__quantity += amount

    def stock_out(self, amount):
        if amount <= self.__quantity:
            self.__quantity -= amount
        else:
            raise ValueError("Insufficient stock")

    def is_expired(self):
        return date.today() > self.expiry_date

    def is_low_stock(self):
        return self.__quantity < 10

    def get_quantity(self):
        return self.__quantity

```

Inventory.py



```python
class Inventory:
    def __init__(self):
        self.drugs = {}

    def add_drug(self, drug):
        self.drugs[drug.batch_number] = drug

    def get_drug(self, batch_number):
        return self.drugs.get(batch_number)

    def all_drugs(self):
        return list(self.drugs.values())
```

User.py

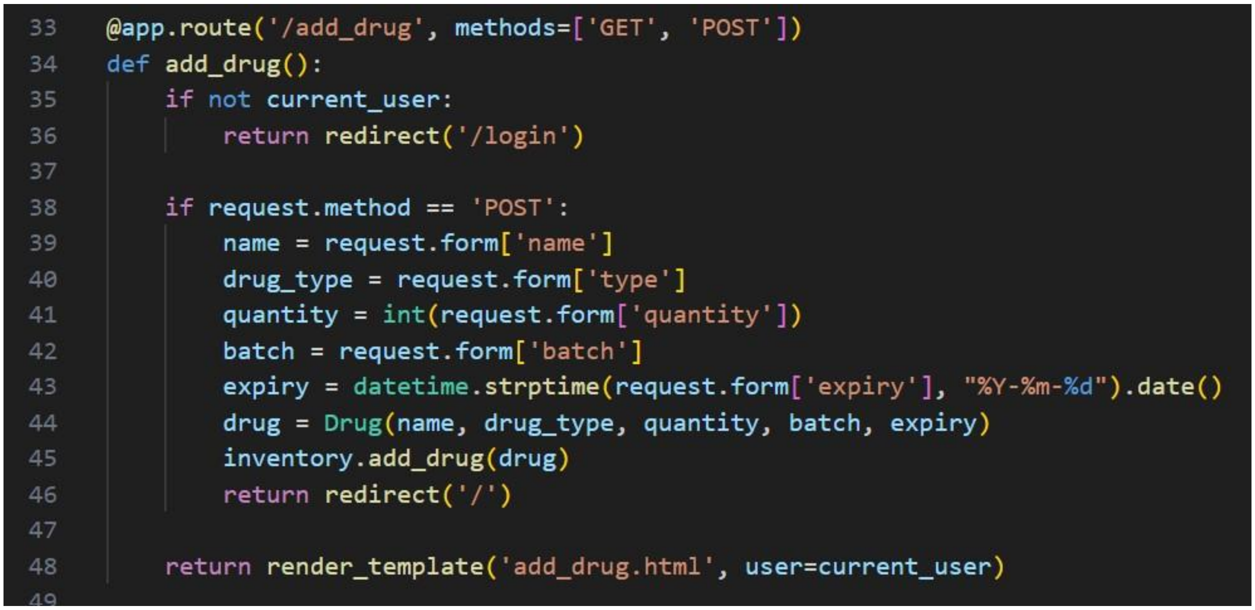


```python
class User:
    def __init__(self, username):
        self.username = username

    def get_role(self):
        return "User"

class Pharmacist(User):
    def get_role(self):
        return "Pharmacist"
```

Flask app implementation

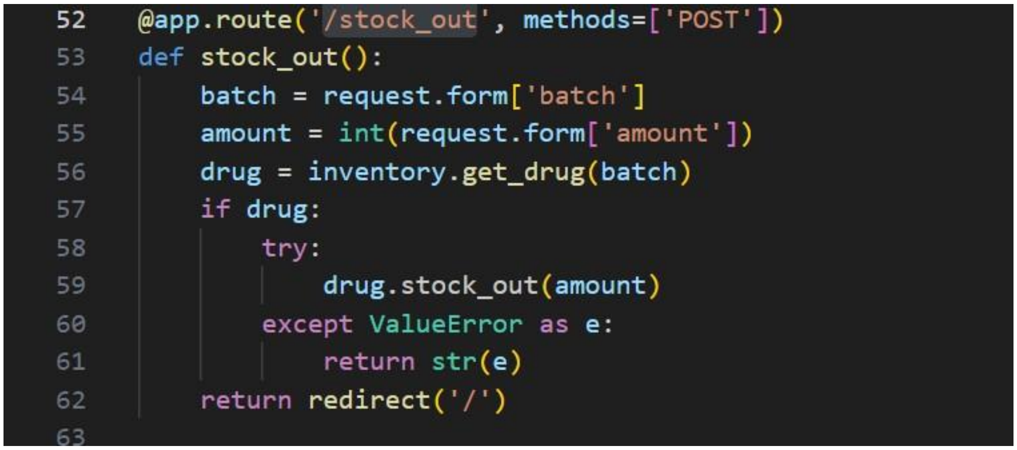
The main logic of the system is in app.py which routes requests to the appropriate functions and views

For example, a route for /add_drug



```python
@app.route('/add_drug', methods=['GET', 'POST'])
def add_drug():
    if not current_user:
        return redirect('/login')

    if request.method == 'POST':
        name = request.form['name']
        drug_type = request.form['type']
        quantity = int(request.form['quantity'])
        batch = request.form['batch']
        expiry = datetime.strptime(request.form['expiry'], "%Y-%m-%d").date()
        drug = Drug(name, drug_type, quantity, batch, expiry)
        inventory.add_drug(drug)
        return redirect('/')

    return render_template('add_drug.html', user=current_user)
```

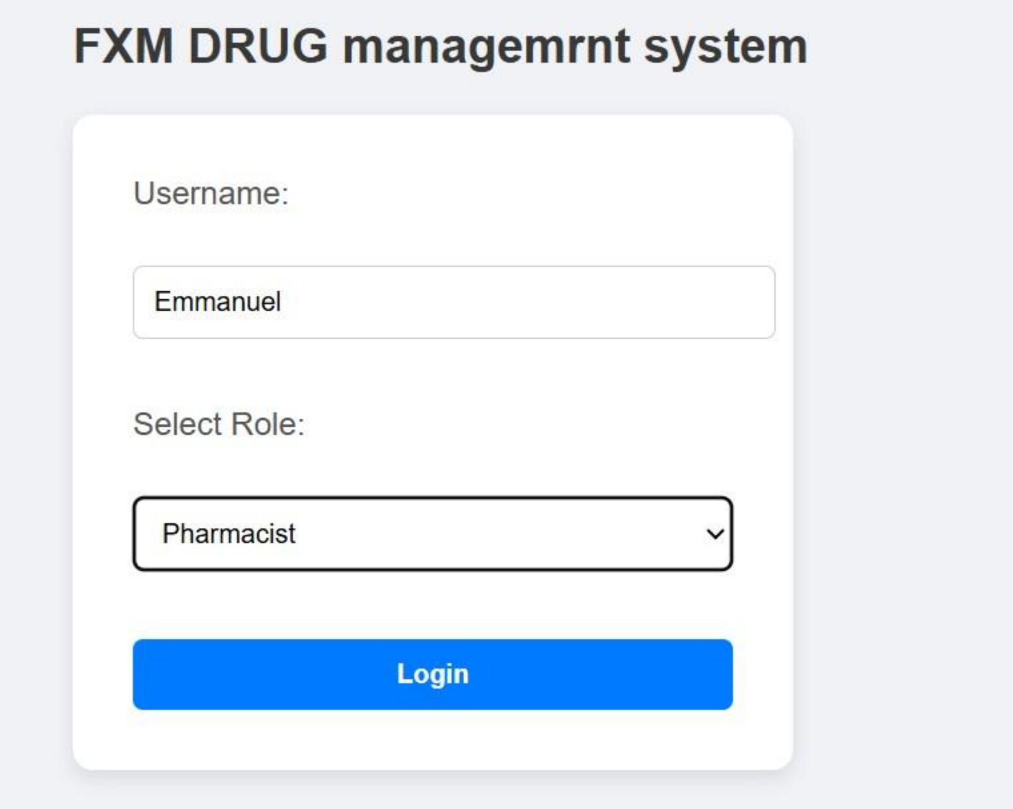
/stock_out



```python
@app.route('/stock_out', methods=['POST'])
def stock_out():
    batch = request.form['batch']
    amount = int(request.form['amount'])
    drug = inventory.get_drug(batch)
    if drug:
        try:
            drug.stock_out(amount)
        except ValueError as e:
            return str(e)
    return redirect('/')
```

Below is how the system functions in the front end

Login in screen



FXM DRUG managemrnt system

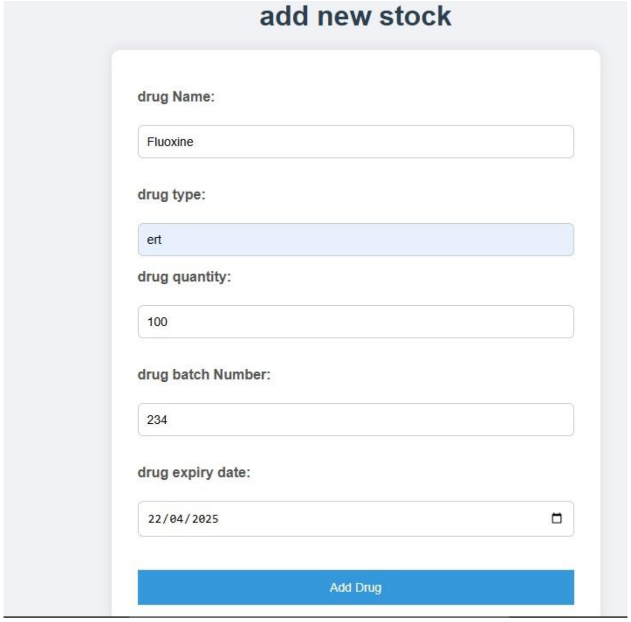
Username:

Emmanuel

Select Role:

Pharmacist

Adding new medicine



add new stock

drug Name:

Fluoxine

drug type:

ert

drug quantity:

100

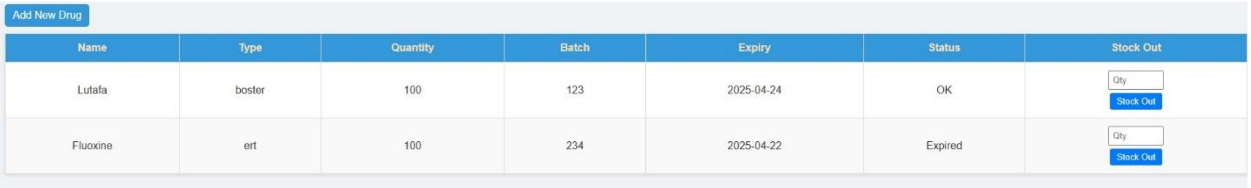
drug batch Number:

234

drug expiry date:

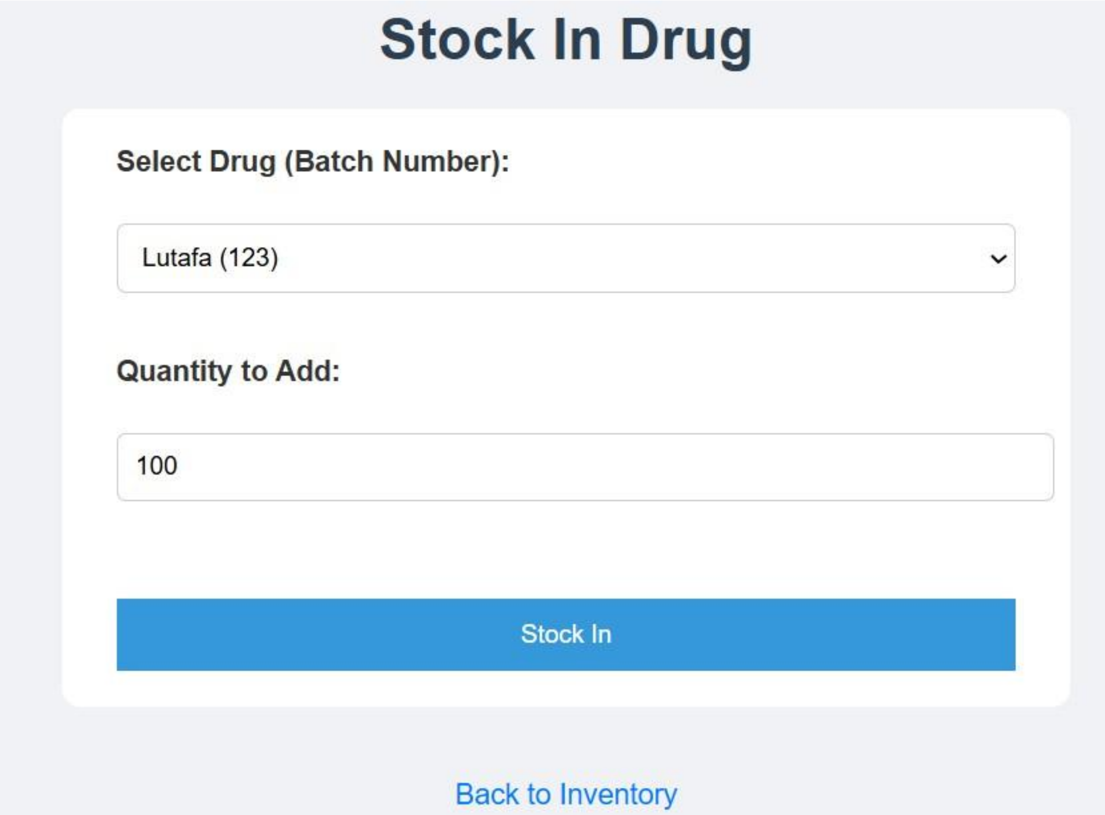
22/04/2025

Added stock



Name	Type	Quantity	Batch	Expiry	Status	Stock Out
Lutafa	boster	100	123	2025-04-24	OK	Qty Stock Out
Fluoxine	ert	100	234	2025-04-22	Expired	Qty Stock Out

Add stock in to already existing

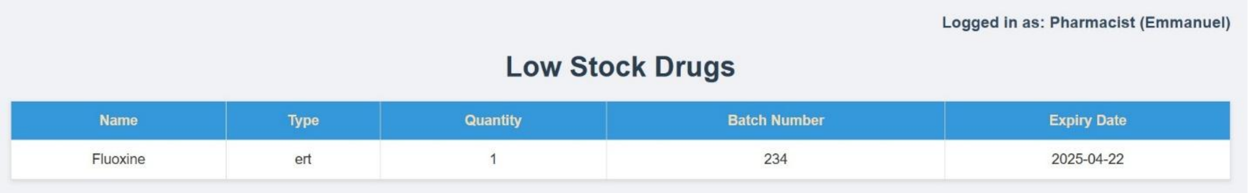


Stock In Drug

Select Drug (Batch Number): Lutafa (123)

Quantity to Add: 100

Viewing medicine with low sock after giving out some



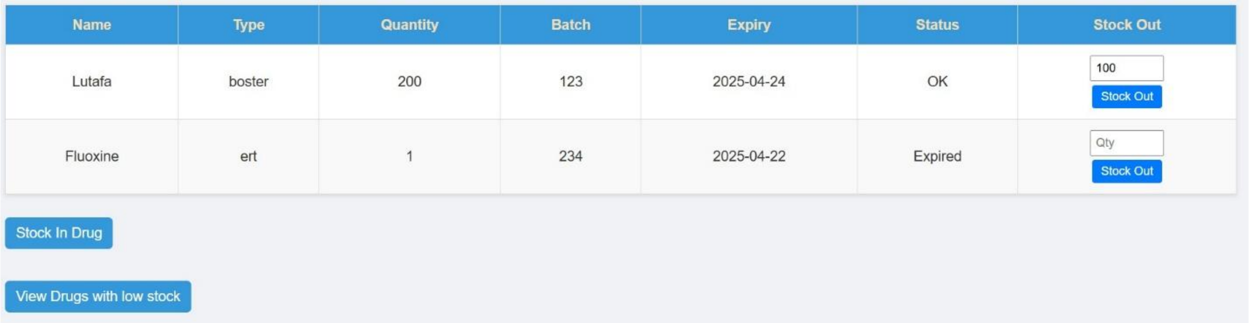
Logged in as: Pharmacist (Emmanuel)

Low Stock Drugs

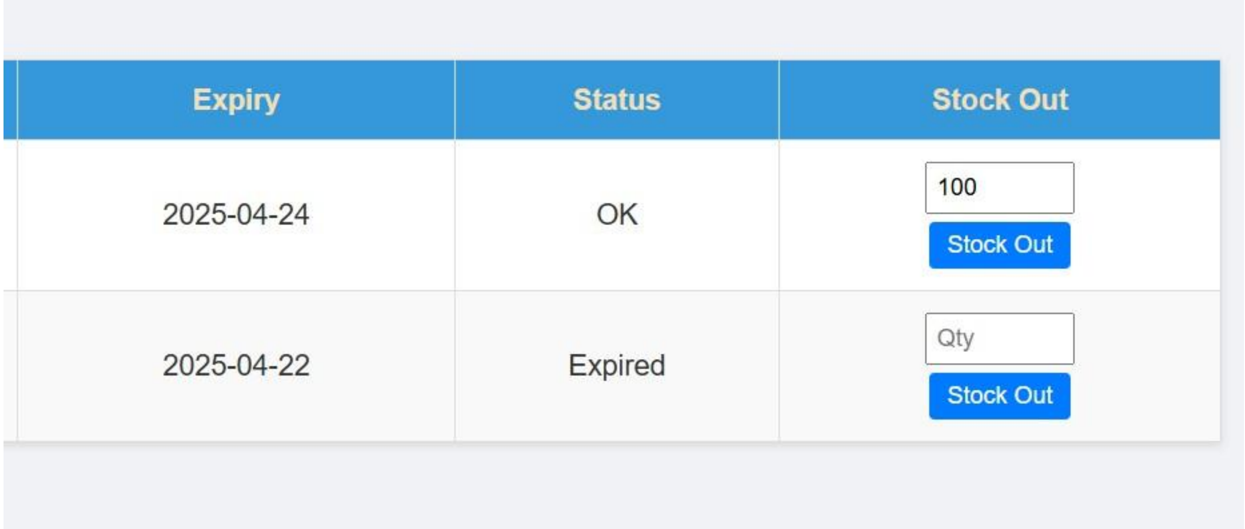
Name	Type	Quantity	Batch Number	Expiry Date
Fluoxine	ert	1	234	2025-04-22

Up fluoxine has been deducted by 99

Below is where stock is given out and it also shows the expired stock



Name	Type	Quantity	Batch	Expiry	Status	Stock Out
Lutafa	boster	200	123	2025-04-24	OK	100 Stock Out
Fluoxine	ert	1	234	2025-04-22	Expired	Qty Stock Out



Expiry	Status	Stock Out
2025-04-24	OK	100 Stock Out
2025-04-22	Expired	Qty Stock Out

When you observe the above 2 figures you see that fluoxine is expired

Also, in the last column is where stock outs are made.