https://github.com/Musemma100/drug-inventory-management

**System design**

The system will be developed consisting of the frontend, backend and the in-memory storage

So bellow are them

**2.1 Frontend**

the frontend is designed using HTML and basic CSS. Its includes the following user interfaces  
A login form for simulating the user access

A dashboard to view the drug 4inventory

It has forms for: Adding new drugs

Stocking in more drugs

And viewing low stock drugs

Thes front end templates are located in the templates folder of the project

**2.2 Backend :**

The backend is built using flask(python. It is handling the systems business logic and routes. The key responsibilities include

* Registering and managing drugs
* Performing stock in/ out operations
* Validating expiry dates
* Filtering low stock drugs
* Simulating user roles using OOP concepts

All data is stored in memory during the app runtime using python date structures

**2.3 System components**

**2.3.1 Drug class**

Represents a medicine with key attributes and encapsulated logic

Encapsulation: the attribute quantity is private and modified through methods

Abstraction: methods like is\_expired() and is\_low\_stock() abstract internal logic

Drug

* -name: str
* -drug\_type: str
* -\_\_quantity: int
* Batch\_number: str
* -expiry\_date: date
* +is\_expired(): bool
* +is\_low\_stock(): bool
* +stock\_in(amount: int): void
* +stock\_out(amount: int): void

**User and pharmacist classes:**

Used to simulate the different roles in the system:

Inheritance: pharmacists inherits from the base user class.

Polymorphism: the get\_role() method behaves differently depending on the user type.

User

* -username: str
* + get\_role(): str
* Pharmacist(inherits from user
* + get\_role(): str

**Inventory class**

Thia class manages all registerecd drugs in memory , it supports adding, retrieving and listing all drugs

Drugs: dict

* + add\_drug(deug: drug): void
* + get\_drug(batch: str): drug
* + list\_all((): list

**System structure**

exam\

|-app.py (Flask app)

|--models\

| |--drug.py (drug class)

| |--user.py user abd phamacist

| |--inventory.py (inventory management)

|--templates\

| |--index.html home page

| |--add\_drug.html add drugs form

| |--stock\_in.html stocking form

| |--login.html login form

| |--low\_stock.html low stock lists

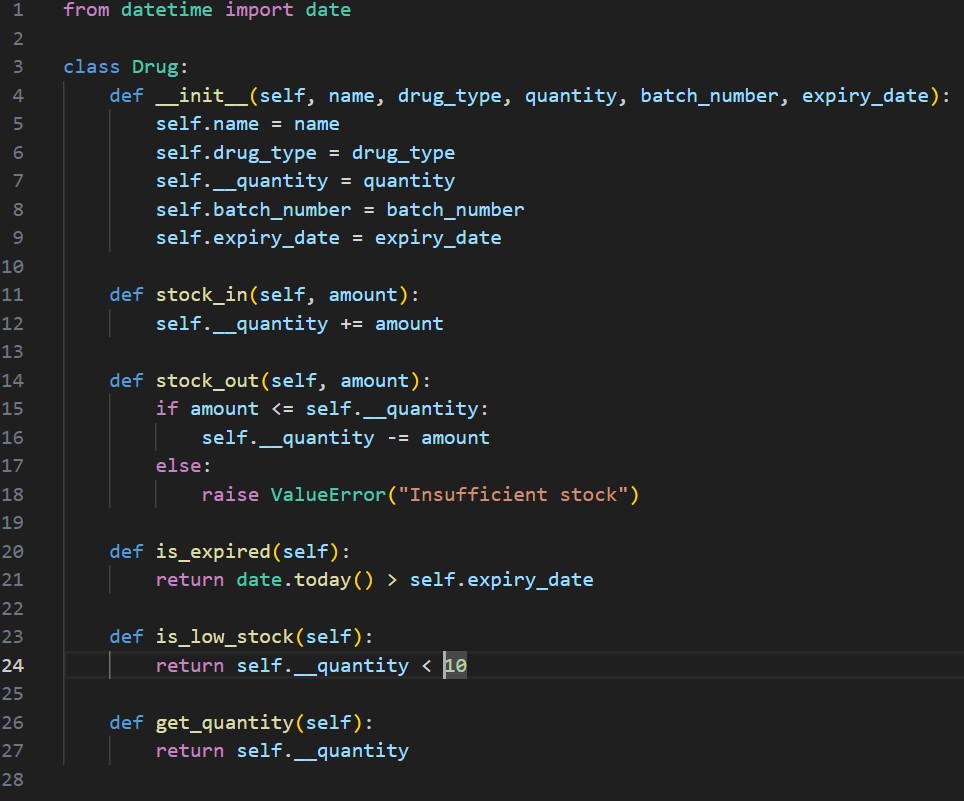
**3. System implementation**

This section describes how the drug inventory management system was implemented using python mostly using Flask and the OOP principles, . the system was developed following a modular structure so as to archive flexibility and maintability.

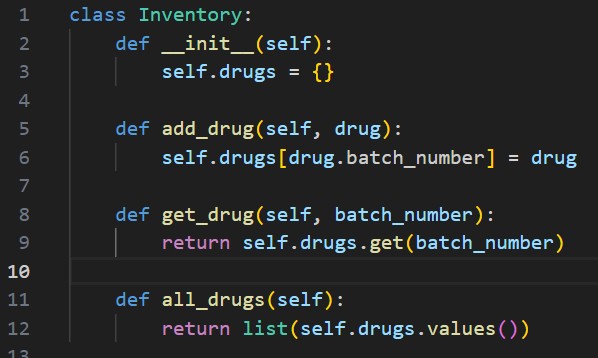
Technologies used included python. Flask was used to develop the web framework. The front end was developed using HTML. The logic design was designed using python classes.

Back end implementation was done using flasks shown below

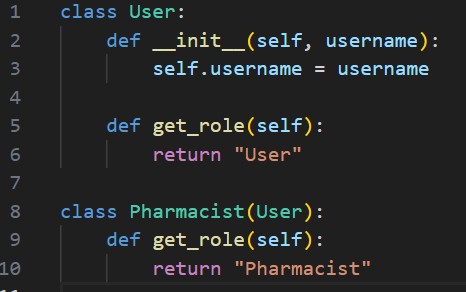
**Drug.py**



Inventory.py



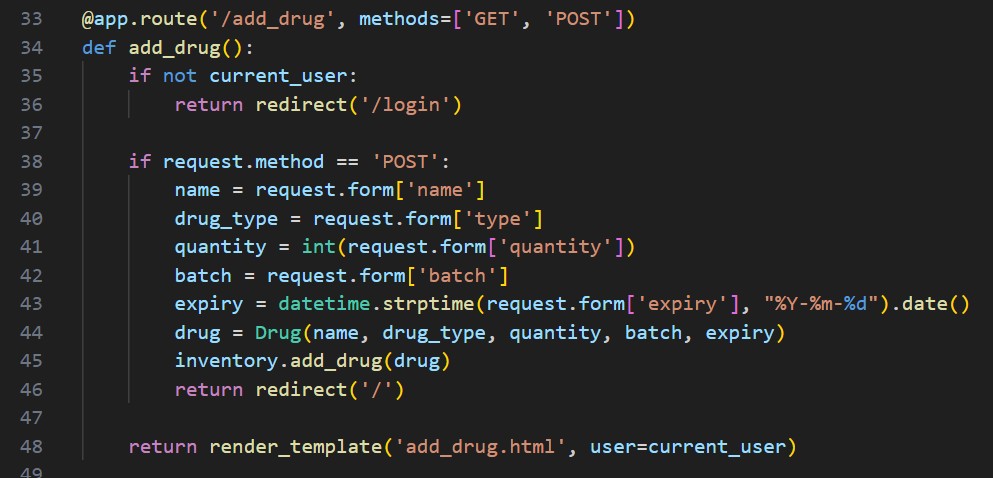
User.py



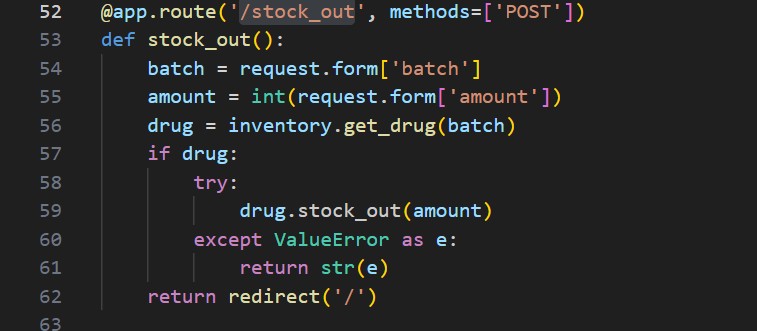
Flask app implementation

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

For example, a route for /add\_drug



/stock\_out



Frontend implementation

The user interacgts with thev system through the HTML forms . each form page for example add drug, login and stockin is rendered usijg render\_template() in flask and passed context variables likr the current user or inventory list.