International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 6, Nov-Dec-2026, ISSN (Online): 2395-566X

An Intelligent System for Pharmaceutical Inventory Management Based on Demand Forecasting and Expiry-Risk Prediction

Dr. Pradeep[1], Dr. Kirti[2], Nilesh Paliwal[3], Aditya Solanki[4], Kanika Khurana[5]
[1,2,3,4,5] department Of Computer Science And Engineering, Hmritm, Hamidpur, New Delhi 110036, India

Abstract: Managing a pharmaceutical inventory requires holding enough medicine to meet needs while avoiding surplus purchases, since any stock that outlives its expiry date must be discarded. Most inventory software stores quantities and expiry dates but cannot indicate whether a particular batch will be sold before it expires. This paper presents a decision-support framework that predicts weekly demand for every medicine, applies the prediction to estimate how much of each batch will still be unsold on its expiry date, and converts these estimates into reorder suggestions and expiry alerts. A naïve forecast, an 8-week moving average and a Random Forest were compared, and the complete framework was evaluated in a weekly inventory simulation using 100 randomly generated datasets of 15 medicines each. Real batch records were unavailable, so every dataset is synthetic; the findings illustrate how the method behaves and do not measure the savings a real pharmacy would obtain. The moving average produced the smallest one-week-ahead error (WAPE 25.2%), whereas the Random Forest was more accurate for total demand over 12 weeks (19.9% versus 22.9%). When batches were examined 12 weeks before expiry, the forecast-based rule issued 109 alerts for 1,563 batches, compared with 1,563 alerts from a date-only rule, and achieved 89% precision and 83% recall for batches that finished with at least one fifth of their stock unsold. Capping order quantities according to expiry risk cut average stock value by roughly 5%, yet it did not significantly reduce expiry losses and it lowered the fill rate for slow-moving medicines. The alerts were therefore the more valuable component of the framework in this study. We also built a web prototype around the framework using Python and Flask, a SQLite or PostgreSQL database, role-based login, batch-wise stock, billing and a dashboard. Only the operation of this prototype was verified, and its outputs are excluded from the experiments.

Keywords: pharmaceutical inventory control; medicine demand forecasting; expiry-risk prediction; machine learning; inventory optimisation; decision support system; Flask web application

I. INTRODUCTION

Hospitals, pharmacies and medical stores must have medicines on hand at all times, yet they cannot afford to purchase more than will be consumed. Because medicines carry an expiry date, stock remaining after that date represents a financial loss and also requires safe disposal. Getahun et al. [1] studied expired medicines in nine public hospitals of Jimma Zone, Ethiopia, and reported an average wastage rate of about 4.87% across two financial years, more than double the national target of 2%. The contributing factors they identified include the delivery of stock close to expiry, irrational prescribing and limited involvement of clinicians in choosing and quantifying medicines. The opposite problem is understocking: when a medicine is unavailable, a patient may not be treated on time.

Most deployed inventory systems record the medicine name, batch number, quantity, supplier and expiry date. They describe what is currently in the store but say little about what is likely to happen next. A pharmacist needs answers to two questions. The first is how much of each medicine will be required over the coming weeks. The second is whether the stock already on the shelf will be consumed before it expires. A medicine may hold sufficient stock in total and still contain a single batch that will not be finished in time.

Demand forecasting for medicines has been approached with statistical as well as machine learning methods. Zhu et al. [2] combined supply-chain information with machine learning to forecast pharmaceutical demand, and Pall et al. [3] used pharmacy data to anticipate drug shortages. Further work has compared machine learning against traditional methods for essential medicines [4], [5]. Most of these studies end at the forecast: the output is a quantity in units, and the user must still decide what to order. Machine learning also does not always yield better forecasts. Makridakis et al. [6] observed that well-known machine learning methods were less accurate than simple statistical ones on a large collection of monthly series, which means that any new model should be benchmarked against simple baselines.

A separate research stream in operations research addresses perishable inventory, for which Nahmias [7] offers a good starting review. That literature examines ordering and issuing rules for items with a limited life, but it normally assumes a known demand distribution and does not treat the individual

© 2026 IJSRET

batches held in a pharmacy store. We found few studies that link a demand forecast to the expiry date of each batch in a form a pharmacy could use directly.

This paper attempts to bridge the two streams. Our contributions are as follows. First, we describe a framework in which the demand forecast is used to estimate how much of each batch is likely to remain unsold at expiry, while accounting for the order in which batches are consumed. Second, we compare three forecasting models and five inventory policies in a weekly simulation that includes batch expiry. Third, we present a web prototype (Python and Flask back end, SQLite or PostgreSQL database, browser dashboard) that supplies the inventory, batch, billing and analytics functions surrounding the framework. Fourth, we report which approaches worked and which did not, including outcomes that differed from our expectations. The data are synthetic because batch-level records from a real store were not accessible, which restricts the conclusions that can be drawn. The prototype was checked only for working operation and is excluded from the quantitative evaluation.

The remainder of the paper is arranged as follows. Section II reviews related studies. Section III states the problem, objectives and research questions. Section IV explains the methodology, and Section V describes the system design. Section VI sets out the experimental design, Section VII reports the results, and Section VIII concludes.

II. LITERATURE REVIEW

1. Pharmaceutical Demand Forecasting

Zhu et al. [2] investigated pharmaceutical demand forecasting by pairing machine learning with supply-chain information. They noted that past sales are not the only informative signal and that data such as downstream inventory can improve predictions. This matters for our work because it shows that knowledge of stock levels can strengthen a forecast. X. Chen et al. [8] merged a knowledge graph with an LSTM network so that relationships among medicines are used alongside each medicine's own history. Models of this type can be accurate, but they demand more data and computing capacity than a small pharmacy is likely to possess.

Several studies have applied standard machine learning models to medicine demand. Mbonyinshuti et al. [4] employed a random forest to predict the demand for essential non-communicable disease medicines in Rwandan public health facilities. A recent study contrasted traditional statistical techniques with machine learning for essential medicines, using data from the EPSS Gondar hub in Ethiopia [5]. Fourkiotis and Tsadiras [9] applied both machine learning and statistical forecasting to pharmaceutical sales. Collectively, these studies indicate that multiple models should be tested on the same data rather than presuming that one model fits every medicine.

The broader forecasting literature conveys a similar message. In an analysis of 1,045 monthly series from the M3 competition, Makridakis et al. [6] found the machine learning methods they examined to be less accurate than traditional statistical methods at every horizon. In the later M5 competition, which used retail sales, the leading entries were machine learning methods [10]. The outcome therefore depends on the data. Pharmacy demand is also frequently intermittent, with many weeks of zero or very low sales for slow-moving items. Syntetos et al. [11] proposed classifying such demand by the mean interval between demands and the variability of demand size, and showed that the most suitable forecasting method differs between categories. We compare models using WAPE. Hyndman and Koehler [12] showed that many common accuracy measures fail in situations that arise often, such as actual values of zero, which is relevant for slow medicines.

2. Drug Shortage And Inventory Prediction

Pall et al. [3] predicted drug shortages from pharmacy data using machine learning and also classified shortage severity. Predicting shortages differs from predicting expiry, but both aim to alert the pharmacy early enough for action to be taken.

Some recent studies link forecasting to the ordering decision. Galli et al. [13] proposed a prescriptive analytics method for hospital pharmacies that draws on ward-level patient and dosing data. J. Chen et al. [14] constructed a neural network for hospital drug inventory that is trained to output order quantities directly. Its loss function applies different penalties to overstocking and understocking, and on hospital data from Beijing it achieved a lower cost than the conventional two-step approach of forecasting first and optimising afterwards. Goltsos et al. [15] discussed the divide between forecasting research and inventory research. We retain the two-step structure in this paper because it is simpler to explain and implement, but the central message of these studies is valuable to us: a more accurate forecast does not by itself guarantee a better inventory decision. We return to this point in Section VII.

3. Expiry-Aware Inventory Management

The classic reference on perishable inventory is the review by Nahmias [7], which treats items with a fixed life as well as items that decay continuously, and also covers blood banks. Pierskalla and Roach [16] examined the sequence in which stock should be issued and concluded that, under their assumptions, issuing the oldest stock first is optimal. In a pharmacy, the corresponding rule is to issue the batch with the earliest expiry date first, known as FEFO. Karaesmen et al. [17] subsequently surveyed research on perishable and ageing inventories and proposed directions for further study.

© 2026 IJSRET

More recent studies have begun to use forecasts directly. Motamedi et al. [18] analysed a finite-horizon, periodic-review system for perishable products in which demand forecasts are available. They demonstrated that, given certain assumptions about the forecasts, the optimal policy is a base-stock level that depends on current inventory, the forecasts and past demand. Ketzenberg et al. [19] examined how the expiration date and the order quantity should be chosen jointly for a product with a random lifetime.

These studies offer useful ideas for inventory control, but consumer perishables differ from medicines, which are subject to regulatory requirements for storage and handling. In our system, the recorded expiry date of a batch is accepted as given, and we do not attempt to alter or predict the official expiry date of any medicine.

4. Summary Of Literature

The literature supplies forecasting methods for medicines, a handful of decision frameworks for hospital pharmacies and an extensive theory of perishable inventory. What we did not find is a simple batch-level calculation that takes a demand forecast, the current stock of each batch and the issuing order, and returns a risk value that a pharmacist can act on early. This paper aims to close that gap and tests the idea by simulation.

III. PROBLEM STATEMENT AND OBJECTIVES

1. Problem Statement

A pharmaceutical inventory must keep medicines available while restricting surplus stock and losses from expiry. Conventional systems mostly support the recording and monitoring of stock, and they may offer too little predictive support for estimating future demand or for identifying batches likely to stay unused until they expire. This study therefore proposes an integrated pharmaceutical inventory management system that joins demand forecasting with expiry-risk assessment in order to provide well-founded replenishment and stock-control recommendations.

2. Research Objectives

- To build a digital system for managing pharmaceutical inventory.
- To examine historical records of medicine sales or consumption.
- To assess statistical and machine learning models for demand forecasting.
- To estimate the expiry risk of each batch from its remaining shelf life, available stock and forecast demand.
- To produce inventory recommendations, including replenishment alerts and expiry warnings.
- To assess the proposed framework with forecasting and inventory performance measures.

3. Research Questions

- **RQ1:** How accurately can the future demand for medicines be estimated from historical consumption records?
- **RQ2:** How does a machine learning forecasting model compare with simple statistical baselines?
- **RQ3:** Can an expiry assessment based on forecasts identify batches that are likely to remain unused until expiry?
- **RQ4:** Does combining demand forecasts with expiry information lead to better inventory decisions in a simulated or observed setting?

IV. PROPOSED METHODOLOGY

1. System Architecture

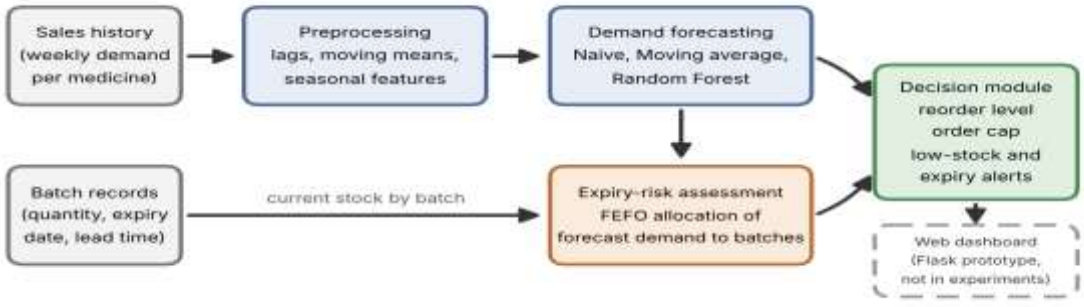


Fig. 1. Architecture of the proposed system. The dashed box (web dashboard) represents the Flask prototype of Section V and was not part of the experiments.

© 2026 IJSRET

The proposed system consists of five components: data management, data preprocessing, demand forecasting, expiry-risk assessment and inventory decision support (Fig. 1). Sales and batch records are processed to generate demand forecasts and expiry-risk estimates, which are merged into inventory recommendations and displayed to the user on a management dashboard. In this study, the forecasting, expiry-risk and decision components were implemented in Python and evaluated by simulation. The management dashboard belongs to the web prototype described in Section V and was not used in the experiments.

2. Dataset Description

The system requires historical medicine consumption data together with batch-level inventory details. Sales data should include the date, medicine identifier, quantity sold and, where available, the unit price and store identifier. Batch data should include the medicine identifier, batch identifier, available quantity, manufacturing date, expiry date and supplier lead time.

Real pharmacy data containing batch records was unavailable to us, so synthetic data were generated, and all results in this paper should be interpreted accordingly. For each of 100 replications we generated 208 weeks of demand for 15 medicines, comprising five fast-moving, five medium-moving and five slow-moving items (Table I). Weekly demand follows a yearly seasonal pattern and a linear trend. Random variation was introduced through a gamma-Poisson (negative binomial) distribution, which reproduces the overdispersion and the numerous zero-demand weeks typical of slow medicines. In the final year, roughly one third of the medicines experience an abrupt fall in demand (to 40–70% of the earlier level) and about one in six experiences an abrupt rise (by 30–60%). These shifts are intended to resemble a change in prescribing habits or a local outbreak, and the models have no advance knowledge of them. All settings in Table I were selected by us to resemble typical pharmacy demand and were not fitted to any real data.

For batches, each order is delivered after the medicine's lead time in a pack size determined by its class, and every delivered batch has between 12 and 60 weeks of remaining shelf life, drawn at random. Slow-moving medicines come in large packs relative to their weekly sales and carry a higher unit cost, as is common for specialist drugs. The code can also read a real sales file (one date column and one column per medicine), but no real data were used for the results reported here.

TABLE I. Parameters used to generate the synthetic data. The negative binomial dispersion parameter k was 20, 8 and 3 for fast, medium and slow medicines, respectively. The trend over four years ranged from −15% to +25%. Costs are in arbitrary currency units.

Class	Items	Base demand (units/week)	Seasonal amplitude	Lead time (weeks)	Pack size	Unit cost
Fast	5	150–400	0.10–0.40	1	about 1 week of demand	1–10
Medium	5	30–100	0.10–0.50	1–2	2–4 weeks of demand	5–30
Slow	5	3–12	0.00–0.40	2	15–35 weeks of demand	20–150

3. Data Preprocessing

When real data are used, records should first be screened for missing values, duplicates, inconsistent medicine identifiers and invalid dates; this step was unnecessary for the generated data. Sales are aggregated into weekly totals. The data are split chronologically, with the first 156 weeks used for training and the final 52 weeks for testing, so that no model is trained on weeks that follow those it is tested on.

The Random Forest uses the following features: demand in each of the previous four weeks, the mean of the last 4 and the last 13 weeks, the mean demand around the same week one year earlier (weeks t−53 to t−51), and the week of the year encoded as a sine and a cosine. Demand is divided by the training-period mean of the corresponding medicine, which allows a single model to be trained across all medicines. Forecasts beyond one week ahead are generated recursively, meaning that the model's own earlier forecasts serve as inputs for later weeks, up to 24 weeks.

4. Demand Forecasting

Three approaches were compared: (1) a naïve forecast, which uses the most recently observed demand; (2) a moving average of the last 8 weeks; and (3) a Random Forest regressor [20] that estimates future demand from historical demand and the features above. The Random Forest was built with scikit-learn [21] using 100 trees, a minimum of 3 samples per leaf and 60% of the features considered at each split. These settings were not tuned.

All models were assessed on the same test period using mean absolute error (MAE), root mean squared error (RMSE) and weighted absolute percentage error (WAPE). Since the ordering rule needs forecasts for 2 to 3 weeks and the expiry module needs them for up to 12 weeks, we also measured the

© 2026 IJSRET

accuracy of total demand over the next 4 and 12 weeks. The model for each part of the system was then selected on the basis of these results (Section VII).

5. Expiry-Risk Assessment

The expiry-risk module estimates how much of a batch may remain unused when it reaches its recorded expiry date. Let Qb denote the quantity currently held in batch b, rb the number of weeks remaining until it expires, and F(h) the forecast demand for the week beginning h weeks from now. The forecast demand across the remaining life of the batch is Cb = F(0) + F(1) + … + F(rb − 1). When a medicine has several batches, those expiring earlier are consumed first (FEFO), so the demand remaining for batch b is $\hat{D}b$ = max(0, Cb − the total quantity of all batches expiring before b). The estimated leftover quantity is

$$Lb = \max(0, Qb - \hat{D}b) \quad (1)$$

For batches with Qb > 0, the potential leftover proportion is

$$Rb = Lb / Qb \quad (2)$$

Each batch is assessed once, in the first week in which rb is 12 weeks or less, and an expiry alert is raised when Rb ≥ 0.20. The 12-week window (chosen to give the pharmacy time to react) and the 0.20 threshold were fixed before the experiments were run and were not tuned. This is a forecast-based risk calculation and not a supervised machine learning classifier. It has one weakness: it presumes that batches expiring earlier are used up completely before batch b, which does not hold when demand falls below the forecast.

6. Inventory Decision Module

The decision module combines predicted demand, current stock, supplier lead time and expiry-risk information. It produces reorder recommendations when projected stock is insufficient, low-stock warnings, expiry-risk alerts, and a recommendation to dispense batches approaching expiry first in line with the First Expiry, First Out (FEFO) principle.

In the simulation, stock is reviewed every week. With lead time L weeks and review period R = 1 week, the order-up-to level is

$$S = \Sigma(h = 0 \text{ to } L+R-1)\ F(h) + z \cdot \hat{\sigma} \cdot \sqrt{(L + R)} \quad (3)$$

where z = 1.645 (a 95% service target) and $\hat{\sigma}$ is the standard deviation of the model's most recent 26 one-week forecast errors. If the stock on hand plus the stock on order falls below S, the smallest number of packs that restores the position to S or above is ordered.

The expiry-risk module may also restrict the order. It uses the forecast to compute how much stock will remain when the new batch arrives (sarr) and how much demand will exist for the new batch during the 12 weeks of shelf life that suppliers are assumed to guarantee at delivery, A = max(0, F(L) + … + F(L+11) − sarr). The number of packs is limited to

$$nmax = \lfloor\ A / ((1 - \alpha) \cdot \text{pack size})\ \rfloor \quad (4)$$

with α = 0.25, meaning that at most a quarter of the new batch may be left over. If nmax equals zero, one pack is still ordered when the stock expected at arrival is no more than one week of forecast demand, so that the cap does not deliberately create stock-outs. Five policies were compared (Table II). The recommendations are advisory only; the system does not permit expired medicines to be dispensed and does not modify official expiry information.

TABLE II. The five inventory policies compared in the simulation

Policy	Forecast used for reorder level	Issuing rule	Expiry-risk order cap
A (baseline)	Moving average (8 weeks)	FIFO (oldest received first)	None
B	Moving average (8 weeks)	FEFO	None
C	Random Forest	FEFO	None
D	Moving average (8 weeks)	FEFO	Yes, using Random Forest forecast
E	Random Forest	FEFO	Yes, using Random Forest forecast

System Design And Implementation

7. Software Architecture

The system is organised in two layers. The research layer is the Python code described in Section IV, covering preprocessing, the three forecasting models, the batch-level expiry-risk calculation, the decision rules and the simulation. Every result in Section VII originates from this layer. The application layer, referred to here as MEDS, is a web prototype that supplies the everyday functions around it: role-based login, a medicine catalogue, batch intake with expiry dates, a point-of-sale billing counter with invoices, reports and a dashboard. The prototype was adapted from an existing Flask-based pharmacy-operations code base, which the team reviewed, configured and tested; it was not written from scratch.

The prototype follows a three-tier web architecture. The browser front end is a single-page interface built with HTML,

© 2026 IJSRET

CSS and JavaScript, uses Chart.js for dashboard charts, and communicates with the back end through REST requests. The Flask back end is split into four services: authentication (registration, login and role checks using JSON Web Tokens [22] and bcrypt password hashing), inventory (medicine catalogue, batches and outlets), sales (billing, stock deduction and reports) and an AI service. Data are stored through SQLAlchemy in a SQLite file for local use, and a PostgreSQL database can be selected by supplying a connection string. Four user roles exist: Admin, Pharmacist, Warehouse Manager and Finance Manager.

According to the project documentation, the AI service has three components: a scikit-learn linear-regression model that predicts the future unit demand of a medicine from its sales history; an Isolation Forest [23] that flags unusual transactions based on price and quantity; and a replenishment function that proposes a reorder quantity when stock is below predicted demand. It also includes a chat assistant that answers precise questions, such as those about revenue or batches near expiry, using database queries, and forwards general questions to a language model running locally through Ollama. The prototype's linear-regression predictor is not one of the three models compared in Section IV-D, so no claim is made about its accuracy.

When a sale is billed, the prototype deducts stock from the batch with the earliest expiry date, which corresponds to the FEFO rule of Section IV-F (the project documentation calls it FIFO), and it warns about batches expiring within 30 days. This warning depends solely on the date and is therefore the same kind of rule as the date-only rule in Table V. The Random Forest forecast and the batch-level expiry-risk calculation of Section IV are not yet connected to the prototype.

8. Technology Stack

- Front end: HTML5, CSS3, JavaScript (ES6) and Chart.js
- Back end: Python 3.10 or later, Flask and Flask-CORS
- Authentication: PyJWT and bcrypt
- Database: SQLAlchemy with SQLite for development and PostgreSQL for production (selected through the DATABASE_URL variable)
- Machine learning and data processing: scikit-learn, Pandas and NumPy
- Language-model assistant: Ollama running a local model (phi3 or llama3)
- Deployment options in the documentation: Docker, Docker Compose and a WSGI server such as Waitress or Gunicorn
- Research code and figures: Python, Pandas, NumPy, scikit-learn and Matplotlib

The project synopsis originally proposed React.js, Node.js with Express.js and PostgreSQL, together with a separate Python service for forecasting. During development, the team switched to the Python-centred stack above for three reasons. Forecasting and the web service can be written in a single language, which removes the need for an additional interface between two services. Development time was limited, and an existing Flask code base covering the main inventory functions could be adapted. In addition, the previous project guide had recommended a Flask and Python stack.

9. Application Modules

- Medicine Management: maintains the medicine catalogue with price, cost price and category (prototype).
- Batch Management: records batch number, quantity, destination outlet and expiry date at stock intake (prototype).
- Sales and Billing: a point-of-sale counter that checks stock, deducts it from the earliest-expiring batch and issues numbered invoices (prototype).
- Demand Forecasting: naïve, moving-average and Random Forest models (research layer, implemented and evaluated); linear-regression predictor (prototype, documented, not evaluated).
- Expiry Monitoring: batch-level expiry-risk calculation (research layer, implemented and evaluated); 30-day expiry warning (prototype, date-based).
- Inventory Recommendations: reorder and order-cap rules of Section IV-F (research layer); replenishment suggestions (prototype).
- Analytics Dashboard: views of revenue, profit, fast-moving medicines, top medicines and outlet performance (prototype).
- Access Control, Audit and Anomaly Detection: role-based access, an audit-log table and Isolation Forest scanning of transactions (prototype, documented).

10. Prototype Verification And Status

The prototype was run on a Flask development server on port 5000. The application log of 4 November 2026 shows that the front-end files loaded, that a demonstration user was able to register and log in, and that the endpoints for medicines, outlets, batches, customers and sales, the reports on revenue, profit, fast-moving medicines, outlet performance and top medicines, and the replenishment function all returned HTTP 200. The only error in the log is a 404 for the favicon file, which has no effect on the application. The log contains no calls to the demand-prediction, anomaly-detection or chat endpoints.

The database created by the application contains eight tables: users, outlets, medicines, batches, sales, customers, invoices and audit logs. The copy we inspected held two outlets and

© 2026 IJSRET

one user, and no medicine, batch or sales records. These checks therefore show that the endpoints respond, not that the analytics are correct.

We did not measure the accuracy of the prototype's linear-regression predictor or Isolation Forest, did not load real or synthetic sales into it, and did not test it with pharmacy staff. Connecting the Random Forest forecast and the batch-level risk calculation of Section IV to the prototype is the main integration step that remains.

V. EXPERIMENTAL SETUP AND EVALUATION

1. Experimental Setup

The entire experiment was repeated for 100 replications, each with its own randomly generated dataset of 15 medicines. In every replication, the models were trained on the first 156 weeks and evaluated on the final 52 weeks. All models were tested on identical observations. Forecasts are produced at the beginning of each test week using only information available up to the previous week.

The inventory simulation proceeds week by week through the test year. In each week, delivered orders are added as batches, expired batches are removed and counted as waste, the stock review is carried out and an order is placed if required, and then the week's demand is served. Demand that cannot be met is lost rather than backordered. Opening stock is four times the average weekly demand of the last eight training weeks, rounded to the nearest whole pack with a minimum of one pack, and has 30 weeks of shelf life remaining; for slow-moving medicines this is a single pack, which equals 15 to 35 weeks of demand. Stock still on the shelf at the end of the year is not counted as expired. For all five policies, the demand and the shelf lives of delivered batches are identical, so that differences stem from the policy and not from chance. Policies were compared using a paired Wilcoxon signed-rank test across the 100 replications.

To evaluate the expiry alerts, every batch was assessed in the first week in which 12 weeks or less of shelf life remained. A batch is classed as at risk if at least 20% of the quantity present at that moment was still unsold when the batch expired. Only batches with an outcome known within the test year (expired or fully used) were included; the opening stock was not assessed. Three alert rules were compared: a date-only rule that warns for every batch within 12 weeks of expiry, and the risk measure of Section IV-E computed with moving-average forecasts and with Random Forest forecasts. The alerts were evaluated on the runs of policy C, because the alerts themselves do not alter stock.

2. Forecasting Metrics

$$MAE = (1/n)\ \Sigma\ |yi - \hat{y}i| \quad (5)$$

$$RMSE = \sqrt{[(1/n)\ \Sigma\ (yi - \hat{y}i)^2]} \quad (6)$$

$$WAPE = [\Sigma\ |yi - \hat{y}i| / \Sigma\ |yi|] \times 100 \quad (7)$$

Here yi is the actual demand and ŷi the predicted demand. Smaller values indicate smaller forecasting errors, although they depend on the characteristics of the dataset. MAE and RMSE are expressed in units per week and are dominated by the fast-moving medicines, so WAPE is also reported separately for each class. The same WAPE formula is applied to total demand over 4 and 12 weeks, with yi as the actual total and ŷi as the sum of the weekly forecasts.

3. Inventory Evaluation

Inventory performance was measured by the fill rate (the share of demand that was served), the number of unmet units, the number of units that expired, the expiry loss as a percentage of the value of all stock received, the average value of stock held, and the number of orders. For the alerts we used precision (the share of alerts that were correct), recall (the share of at-risk batches that were flagged) and F1. We also checked what share of the leftover units, and of their value, was contained in flagged batches. The proposed policies were compared with a basic inventory-management baseline (policy A). Because the data are simulated, the results are simulation outcomes and not observed reductions in real pharmaceutical wastage.

VI. RESULTS AND DISCUSSION

1. Forecast Accuracy

Table III presents the one-week-ahead accuracy of the three models averaged over the 100 replications. The 8-week moving average had the lowest error on all three measures (WAPE 25.2%), followed by the Random Forest (29.3%) and the naïve forecast (30.5%). The Random Forest outperformed the moving average in only 8 of the 100 replications, and the paired test indicated that the difference is significant ($p < 0.001$). The Random Forest also varied more between replications (standard deviation of WAPE 3.95 versus 1.13), and on average it forecast 4.1% more demand than was actually sold. A probable explanation is that the test year contains abrupt demand drops and trends, which tree-based models cannot extrapolate, but we did not test this directly. Fig. 2 shows the forecasts for one fast-moving and one slow-moving medicine from the first replication.

TABLE III. One-week-ahead forecasting performance on the test year, mean (standard deviation) over 100 replications. MAE and RMSE are in units per week.

© 2026 IJSRET

Model	MAE	RMSE	WAPE (%)
Naïve Forecast	35.07 (4.94)	60.22 (9.72)	30.50 (1.48)
Moving Average	28.93 (3.96)	49.72 (7.90)	25.16 (1.13)
Random Forest	33.46 (5.39)	57.71 (12.39)	29.27 (3.95)

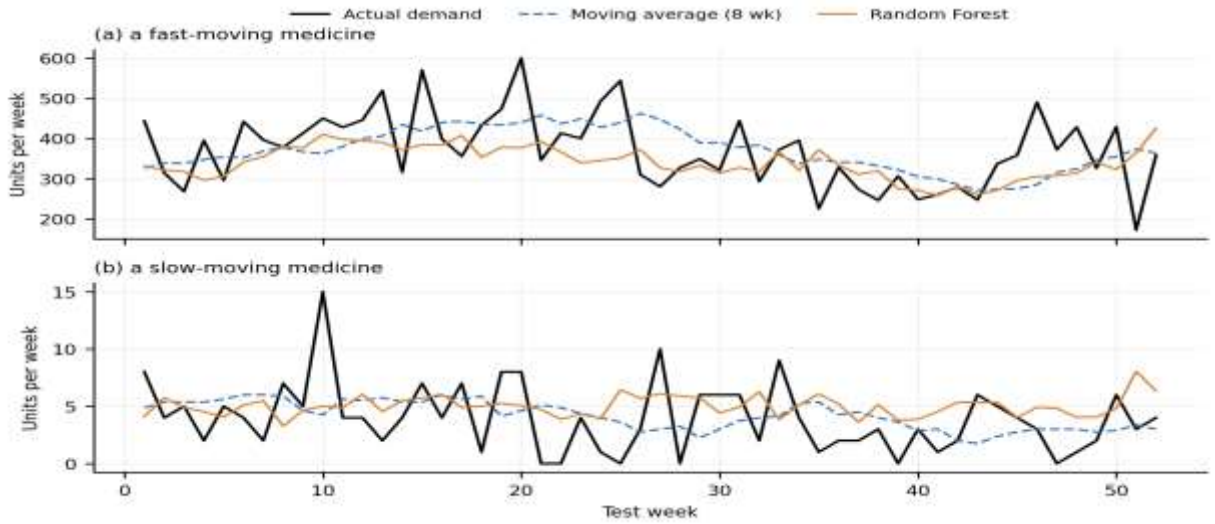


Fig. 2. One-week-ahead forecasts in the test year for (a) a fast-moving and (b) a slow-moving medicine from the first replication

.

Table IV breaks the error down by demand class and also reports accuracy for total demand over longer periods. Errors are far larger for slow-moving medicines (WAPE 57.5% to 74.3%) than for fast-moving ones (22.2% to 26.6%), which is expected because weekly sales of a few units are largely random. The ranking of the models changes at longer horizons. For total demand over the next 4 weeks the moving average remains ahead (18.9% versus 21.3% for the Random Forest), but for the next 12 weeks the Random Forest is the most accurate (19.9%, against 22.9% for the moving average and 26.9% for the naïve forecast), and it beat the moving average in 80 of the 100 replications ($p < 0.001$). This is the horizon that the expiry module requires.

TABLE IV. WAPE (%) by demand class for one-week-ahead forecasts, and for total demand over the next 4 and 12 weeks. Mean over 100 replications.

Model	Fast	Medium	Slow	Next 4 weeks (total)	Next 12 weeks (total)
Naïve Forecast	26.4	42.3	74.3	24.9	26.9
Moving Average	22.2	33.9	57.5	18.9	22.9
Random Forest	26.6	37.3	59.9	21.3	19.9

2. Expiry-Risk Alerts

Across the 100 replications, 1,563 batches were assessed with 12 weeks or less of shelf life and a known outcome. Of these, 117 (7.5%) finished with at least 20% of the assessed stock unsold. The date-only rule captures all 117 but requires 1,563 alerts, so only 7.5% of its alerts were useful (Table V). The forecast-based risk using Random Forest forecasts raised 109 alerts, of which 97 were correct (precision 89.0%, recall 82.9%). The same calculation with moving-average forecasts produced 113 alerts with precision 87.6% and recall 84.6%. The forecast-based assessment thus cut the number of alerts by about 93% while still covering most of the leftover stock. With Random Forest forecasts, the flagged batches contained 91.2% of the leftover units and 93.1% of their value, compared with 89.8% and 89.5% with the moving average. The two forecasting models produced very similar alerts, so in these data the benefit comes more from the batch-level calculation than from the choice of model.

© 2026 IJSRET

TABLE V. Expiry-alert results, all 100 replications together (1,563 batches assessed, 117 of them at risk). Threshold R ≥ 0.20.

Alert rule	Alerts	Correct alerts	Precision (%)	Recall (%)	F1	Leftover value flagged (%)
Date only (12 weeks or less to expiry)	1,563	117	7.5	100.0	0.14	100.0
Forecast risk, moving average	113	99	87.6	84.6	0.86	89.5
Forecast risk, Random Forest	109	97	89.0	82.9	0.86	93.1

3. Inventory Policies

Table VI compares the five policies. Policies A and B, which differ only in the issuing rule, produced nearly identical results, so FIFO versus FEFO made little difference here. A probable reason is that batches of the same medicine rarely remain together on the shelf for long. Policy C, which uses the Random Forest for ordering, had a lower fill rate than A (99.06% versus 99.29%) and more unmet units, consistent with its higher short-term forecast error. Policies D and E, which add the expiry-based order cap, reduced average stock value by about 5% (from 74.2 to 70.3 thousand for D, which was lower in 99 of 100 replications than B, $p < 0.001$). The fill rate fell by only about 0.03 percentage points relative to the same policy without the cap (B and C), although this small difference was also significant.

Expiry losses did not decline in a statistically significant way. The pooled expiry loss was 1.42% of received value for A and 1.36% for both D and E, and the paired tests for expired units gave p-values between 0.13 and 0.65. Nearly all expiry occurred in the slow-moving class (5.8% of received value in A). In that class the cap lowered average stock value by about 8%, but the fill rate dropped from 98.7% to 97.2% (B versus D) while the expiry loss remained at about 5.7%. Varying α between 0.10 and 0.50 made little difference (pooled expiry loss 1.34% to 1.38%, fill rate about 99.0% in every case).

TABLE VI. Inventory results for the five policies, mean over 100 replications (standard deviation in brackets for fill rate and for pooled expiry loss). All 15 medicines are pooled unless stated. Unmet units and orders are totals for one replication.

Policy	Fill rate (%)	Unmet units	Expiry loss, all (% of value)	Expiry loss, slow (% of value)	Average stock value (thousand)	Orders
A	99.29 (0.34)	651	1.42 (1.17)	5.84	74.2	300
B	99.29 (0.34)	648	1.38 (1.16)	5.66	74.2	300
C	99.06 (0.87)	889	1.35 (1.13)	5.59	74.5	303
D	99.26 (0.33)	677	1.36 (1.07)	5.71	70.3	299
E	99.03 (0.87)	916	1.36 (1.07)	5.73	71.0	302

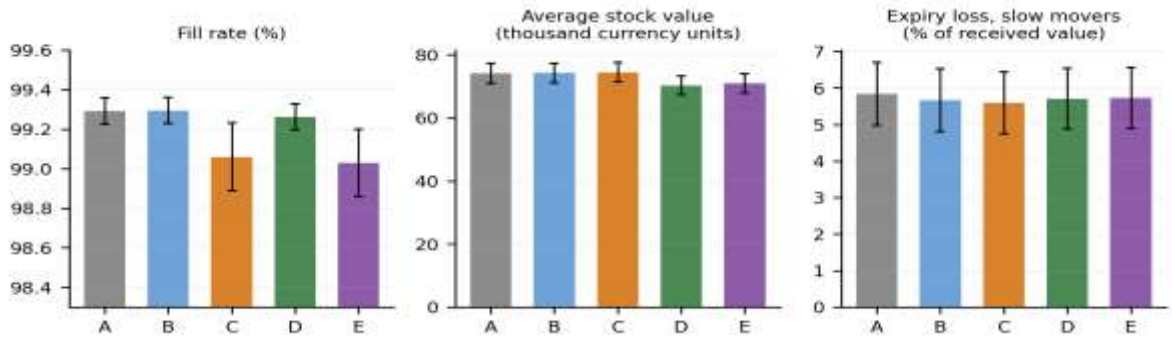


Fig. 3. Fill rate and average stock value (all 15 medicines pooled) and expiry loss of slow-moving medicines for the five policies (mean with 95% confidence interval over 100 replications). The fill-rate axis does not start at zero.

© 2026 IJSRET

4. Discussion

RQ1. With the synthetic data used here, weekly demand could be forecast with a WAPE of about 25–30% across all medicines and about 58–74% for slow-moving ones, and total demand over 12 weeks with an error of about 20–23% for the two better models. These errors arise mostly from the random variation that we built into the data, so they should not be compared with errors obtained on real data in other studies.

RQ2. We expected the Random Forest to be the best model, but at a one-week horizon it was not, and the simple moving average was better in 92 of the 100 replications. The Random Forest was better only at longer horizons. This agrees with the caution of Makridakis et al. [6] that machine learning is not automatically superior to simple methods, while the M5 results [10] show that on other data it can be superior, so the answer depends on the data and on the horizon. Our Random Forest used only each medicine's demand history, about three years of training data and untuned settings, so a tuned model, or one with additional inputs such as holidays or disease data, might perform better. A practical implication is that different parts of the system can use different forecasts: the short-horizon ordering rule performed better with the moving average, and the 12-week expiry check performed better with the Random Forest, although alert quality was almost identical for both.

RQ3. Yes, in this simulation. The batch-level risk calculation identified most of the batches that ended with unsold stock, using far fewer alerts than a date-only rule. This test is easier than real life, because the simulated demand follows the same kind of process that the forecasts assume, and the 20% threshold used to label batches equals the alert threshold. With real data, precision is likely to be lower.

RQ4. The answer is mixed. Using expiry information to cap orders produced a lower average stock value at almost the same service level for the full set of medicines, but it did not significantly reduce expiry losses, and it worsened service for slow-moving medicines. We believe the reason is that, for slow-moving medicines, one supplier pack already covers 15 to 35 weeks of demand while the shelf life on delivery can be as short as 12 weeks, so whether a batch expires depends chiefly on pack size and on the shelf life at delivery, neither of which the ordering rule can change once the order is placed. This is consistent with Getahun et al. [1], where near-expiry stock delivered to hospitals was the factor with the highest agreement among respondents. It is also consistent with the argument of Goltsos et al. [15] and J. Chen et al. [14] that forecast accuracy and decision quality are different matters: the policies with the most accurate short-term forecast (A and B) achieved the best service, and the better 12-week forecast improved the alerts only slightly. An alert is useful only if someone acts on it, for instance by transferring stock to another store, returning it to the supplier or selling it sooner, and we did not simulate these actions.

5. Limitations

The principal limitation is that all data are synthetic and were generated by us, so the results depend on our choices of demand patterns, pack sizes and shelf lives. These values were selected before the experiments to look realistic, but they are not based on measurements from a real pharmacy, and other values could lead to different results. Further simplifications include a single store, a fixed lead time per medicine, lost sales in place of emergency purchases, demand that is independent of the age of the stock, and costs set by us. The Random Forest used nearly default settings and only the sales history. The p-values indicate that a difference is consistent across the simulated replications and say nothing about real pharmacies. Finally, the web prototype was checked only on a development server with a demonstration account and an empty database. Its linear-regression and Isolation Forest modules were not evaluated, the Random Forest and the expiry-risk calculation are not yet connected to it, and its usability was not tested with pharmacy staff.

VII. CONCLUSION AND FUTURE SCOPE

This paper put forward an intelligent pharmaceutical management system that joins demand forecasting and expiry-risk assessment to support inventory decisions. The framework combines weekly medicine demand with current stock and batch-level expiry dates to estimate future demand, identify stock that may expire and generate replenishment recommendations. The forecasting, expiry-risk and decision components were implemented in Python and tested in a weekly simulation with 100 synthetic datasets. The system is also available as a Flask-based web prototype with a SQLite or PostgreSQL database, role-based access, batch-wise stock, billing that dispenses the earliest-expiring batch first, a Chart.js dashboard and an AI service. The prototype was checked through a local test run and is not part of the quantitative evaluation.

In this study, the 8-week moving average was the most accurate model for one-week-ahead demand, while the Random Forest was the most accurate for total demand over 12 weeks. The batch-level expiry-risk calculation reduced the number of alerts by about 93% relative to a date-only rule and identified 83% of the batches that finished with a large portion of their stock unsold, at a precision of 89%. Limiting orders by expiry risk reduced average stock value by about 5% but did not significantly reduce expiry losses. Since the data are synthetic, these results show how the method behaves and are not evidence of savings in a real pharmacy.

© 2026 IJSRET

Future work should first test the framework on real pharmacy or hospital records, or at least on a public sales dataset to which batch records have been added. The forecasting component could be extended with methods for intermittent demand [11], with probabilistic forecasts for safety stock, and with models trained directly for the ordering decision [14]. The expiry component could treat pack size and the minimum shelf life on delivery as decision variables, include stock transfers between stores, and adopt a supervised expiry-risk model once real batch outcomes are available. The prototype should be connected to the Random Forest forecast and the batch-level expiry-risk calculation of this paper, so that its 30-day warning and linear-regression predictor can be compared with them. It should then be tested with real records and pharmacy staff before being deployed with a production database and server or integrated with existing pharmacy management systems.

REFERENCES

1. H. Getahun, S. Belew, G. Hasen, Y. T. Mekasha, and S. Suleman, "Assessment of the extent and monetary loss in the selected public hospitals in Jimma Zone, Ethiopia: Expired medicine perspectives," Front. Med., vol. 11, Art. no. 1283070, 2024, doi: 10.3389/fmed.2024.1283070.
2. X. Zhu, A. Ninh, H. Zhao, and Z. Liu, "Demand forecasting with supply-chain information and machine learning: Evidence in the pharmaceutical industry," Prod. Oper. Manag., vol. 30, no. 9, pp. 3231–3252, 2021, doi: 10.1111/poms.13426.
3. R. Pall, Y. Gauthier, S. Auer, and W. Mowaswes, "Predicting drug shortages using pharmacy data and machine learning," Health Care Manag. Sci., vol. 26, no. 3, pp. 395–411, 2023, doi: 10.1007/s10729-022-09627-y.
4. F. Mbonyinshuti, J. Nkurunziza, J. Niyobuhungiro, and E. Kayitare, "Application of random forest model to predict the demand of essential medicines for non-communicable diseases management in public health facilities," Pan Afr. Med. J., vol. 42, Art. no. 89, 2022, doi: 10.11604/pamj.2022.42.89.33833.
5. T. M. Biadigilign et al., "The prediction of essential medicine demand using machine learning and traditional methods on EPSS Gondar hub 2018–2022 data," Explor. Res. Clin. Soc. Pharm., vol. 23, Art. no. 100801, 2026, doi: 10.1016/j.rcsop.2026.100801.
6. S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "Statistical and machine learning forecasting methods: Concerns and ways forward," PLoS ONE, vol. 13, no. 3, Art. no. e0194889, 2018, doi: 10.1371/journal.pone.0194889.
7. S. Nahmias, "Perishable inventory theory: A review," Oper. Res., vol. 30, no. 4, pp. 680–708, 1982, doi: 10.1287/opre.30.4.680.
8. X. Chen, G. Lu, H. Zhang, and J. Wan, "Knowledge graph-enhanced deep learning for pharmaceutical demand forecasting," Sci. Rep., 2026, doi: 10.1038/s41598-026-35113-4.
9. K. P. Fourkiotis and A. Tsadiras, "Applying machine learning and statistical forecasting methods for enhancing pharmaceutical sales predictions," Forecasting, vol. 6, no. 1, 2024, doi: 10.3390/forecast6010010.
10. S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "M5 accuracy competition: Results, findings, and conclusions," Int. J. Forecast., vol. 38, no. 4, pp. 1346–1364, 2022, doi: 10.1016/j.ijforecast.2021.11.013.
11. A. A. Syntetos, J. E. Boylan, and J. D. Croston, "On the categorization of demand patterns," J. Oper. Res. Soc., vol. 56, no. 5, pp. 495–503, 2005, doi: 10.1057/palgrave.jors.2601841.
12. R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," Int. J. Forecast., vol. 22, no. 4, pp. 679–688, 2006, doi: 10.1016/j.ijforecast.2006.03.001.
13. L. Galli, T. Levato, F. Schoen, and L. Tigli, "Prescriptive analytics for inventory management in health care," J. Oper. Res. Soc., vol. 72, no. 10, pp. 2211–2224, 2021.
14. J. Chen, Z. Xu, and B. Zhu, "Prescriptive analytics for hospital drug inventory: A context-aware multi-channel neural network for integrated forecasting and optimization," Omega, vol. 143, Art. no. 103557, 2026, doi: 10.1016/j.omega.2026.103557.
15. T. E. Goltsos, A. A. Syntetos, C. H. Glock, and G. Ioannou, "Inventory–forecasting: Mind the gap," Eur. J. Oper. Res., vol. 299, no. 2, pp. 397–419, 2022.
16. W. P. Pierskalla and C. D. Roach, "Optimal issuing policies for perishable inventory," Manage. Sci., vol. 18, no. 11, pp. 603–614, 1972.
17. I. Z. Karaesmen, A. Scheller-Wolf, and B. Deniz, "Managing perishable and aging inventories: Review and future research directions," in Planning Production and Inventories in the Extended Enterprise, K. G. Kempf, P. Keskinocak, and R. Uzsoy, Eds. Springer, 2011, pp. 393–436.
18. M. Motamedi, N. Li, and D. G. Down, "Optimal ordering policy for perishable products by incorporating demand forecasts," Eur. J. Oper. Res., vol. 329, no. 1, pp. 124–137, 2026, doi: 10.1016/j.ejor.2025.07.009.
19. M. Ketzenberg, G. Gaukler, and V. Salin, "Expiration dates and order quantities for perishables," Eur. J. Oper. Res., vol. 266, no. 2, pp. 569–584, 2018, doi: 10.1016/j.ejor.2017.10.005.
20. L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

© 2026 IJSRET

21. F. Pedregosa et al., “Scikit-learn: Machine learning in Python,” J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.
22. M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” Internet Engineering Task Force, RFC 7519, 2015, doi: 10.17487/RFC7519.
23. F. T. Liu, K. M. Ting, and Z.-H. Zhou, “Isolation forest,” in Proc. 8th IEEE Int. Conf. Data Mining (ICDM), 2008, pp. 413–422, doi: 10.1109/ICDM.2008.17.

© 2026 IJSRET