

# **CURIOSITY**

## **Medicine Shop Automation**

**Software Requirements Specification**

**PREPARED BY:**

**ARUNDHATI BANERJEE**  
**MEGHNA SENGUPTA**

## Revision History

| Date       | Version | Description | People  |
|------------|---------|-------------|---|
| <25/03/16> | <1.0>   | First draft | Project Owner and Client: Vikas Patidar<br>Faculty Advisor: Dr. Partha Pratim Das<br>Project Group: Arundhati Banerjee<br>Meghna Sengupta |
|            |         |             |   |
|            |         |             |   |
|            |         |             |   |

# Table of Contents

|   |    |
|---|----|
| 1. Introduction                             | 4  |
| 1.1 Purpose                                 | 4  |
| 1.2 Scope                                   | 4  |
| 1.3 Definitions, Acronyms and Abbreviations | 4  |
| 1.4 Overview                                | 6  |
| 2. Customer-Provided Document               | 7  |
| 3. Overall Description                      | 8  |
| 3.1 Product Perspective                     | 8  |
| 3.2 Product Functions                       | 8  |
| 3.3 User Characteristics                    | 9  |
| 3.4 Constraints                             | 9  |
| 3.5 Assumptions and dependencies            | 10 |
| 3.6 Apportionment of requirements           | 10 |
| 4. Specific Requirements                    | 11 |
| 4.1 Use Case Diagram and Documentation      | 11 |
| 4.2 Class Diagram and Documentation         | 25 |
| 4.3 Purchased Components                    | 27 |
| 4.4 Interfaces                              | 27 |
| 4.5 Deployment Diagram                      | 29 |
| 4.6 Licensing Requirements                  | 29 |

# Software Requirements Specification

## 1. Introduction

### 1.1 Purpose

This SRS describes the requirements and specifications of Curiosity, a system for Medicine Shop Automation. It explains the functional features of the system, along with interface details, design constraints and related considerations such as performance characteristics. The SRS is intended for the chemists and druggists of major medicine merchandise.

### 1.2 Scope

The Curiosity MSA is intended to reduce the various laborious tasks of the medicine shop owner. It reduces inventory management overhead by implementing the just-in-time (JIT) philosophy. As much a managerial philosophy as an inventory system, JIT encompasses all activities required to make a final product from design engineering onwards to the last manufacturing operation. JIT systems are fundamental to time based competition and rely on waste reduction, process simplification, setup time and batch size reduction and parallel (instead of sequential) processing.

In addition, the Automation is also capable of handling specialized queries that might be required by the shop owner for efficient management of his stocks. For example, it can generate on being queried, a vendor-wise list of expired medicines which can then be easily replaced. It can also provide accurate figures for revenue and profit for any valid specified period.

### 1.3 Definitions, Acronyms and Abbreviations

#### 1.3.1 *JIT*

Just-in-time or JIT is a methodology aimed primarily at reducing flow times within production as well as response times from suppliers and to customers. Under this philosophy, actual orders dictate what should be manufactured, so that the exact quantity is produced at the exact time as is required. The inventory management system of this MSA system is based on the JIT philosophy.

### **1.3.2 Threshold Value**

Threshold value of any medicine is the quantity of that medicine that should be present in stock at all times to sustain selling for about one week.

The MSA is designed to calculate the weekly average sale of a particular medicine based on the sale records of the previous 4 weeks thus allowing the shop-owner to always adhere to the JIT philosophy.

### **1.3.3 Medicine Description**

A short summary that contains information about the ailments that the medicine can be used against, the dosage of the medicine and the possible side-effects of the medicine. This can also include information about the active and passive ingredients of the drug.

### **1.3.4 Code Number**

A unique number that is generated for every new medicine procured by the shop-owner which he had not dealt in earlier. It is automatically generated by the MSA the first time the medicine shop procures a new medicine. It is also used for reference on all further dealings related to that particular medicine.

### **1.3.5 Generic Name**

Each medicine has an approved name called the **generic** name. A group of medicines that have similar actions often have similar-sounding generic names.

### **1.3.6 Trade Name**

Every medicine also has a **trade** name. This is chosen by the vendor that it has been purchased from. Several vendors may sell the same generic medicine, each with their own trade name. The name is often chosen to be memorable for advertising, or to be easier to say or spell than the generic name.

### **1.3.7 GUI (Graphical User Interface)**

**GUI** is a type of interface that allows users to interact with electronic devices through graphical icons and visual indicators such as secondary notation, as opposed to text-based interfaces, typed command labels or text navigation. As the name suggests, this sort of interface makes the system a lot more user-friendly and makes it easy to use.

The Curiosity MSA is completely GUI based which helps in increasing the owner-system

interactions.

### 1.3.8 OSS (*Open-source software*)

Open-source software (OSS) is computer software with its source code made available with a license in which the Copyright holder provides the rights to study, change, and distribute the software to anyone and for any purpose. Open-source software may be developed in a collaborative public manner. Open-source software is the most prominent example of open-source development.

### 1.3.9 SQL

Structured Query Language (SQL) is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS).

The Curiosity MSA uses SQL for storing the information on medicines and vendors securely.

## 1.4 Overview

The rest of the SRS examines the specifications of the Curiosity Medicine Shop Automation in detail. Section 2 of the SRS presents the document provided by the customer specifying the properties and functions that are required to be implemented. Section 3 of the SRS presents the general factors that affect the Curiosity MSA and its requirements, such as user characteristics and project constraints. Section 4 outlines the detailed, specific functional, performance, system and other related requirements of the Curiosity Medicine Shop Automation.

- This document of Software Requirements Specification for the MSA is intended mainly for :
- ◆ Developers
  - ◆ People involved with System maintenance
  - ◆ Customer ( for the automation software)

## 2. Customer-Provided Document

### Medicine Shop Automation (MSA):

Perform structured analysis and structured design for the following Medicine Shop Automation (MSA) software:

A retail medicine shop deals with a large number of medicines procured from various manufacturers. The shop owner maintains different medicines in wall mounted and numbered racks.

- The shop owner maintains as few inventory for each item as reasonable, to reduce inventory overheads after being inspired by the just-in-time (JIT) philosophy.
- Thus, one important problem the shop owner faces is to be able to order items as soon as the number of items in the inventory reduces below a threshold value. The shop owner wants to maintain medicines to be able to sustain selling for about one week. To calculate the threshold value for each item, the software must be able to calculate the average number of medicines sales for one week for each part.
- At the end of each day, the shop owner would request the computer to generate the items to be ordered. The computer should print out the medicine description, the quantity required, and the address of the vendor supplying the medicine. The shop owner should be able to store the name, address, and the code numbers of the medicines that each vendor deals with.
- Whenever new supply arrives, the shop owner would enter the item code number, quantity, batch number, expiry date, and the vendor number. The software should print out a cheque favoring the vendor for the items supplied.
- When the shop owner procures new medicines it had not dealt with earlier, he should be able to enter the details of the medicine such as the medicine trade name, generic name, vendors who can supply this medicine, unit selling and purchasing price. The computer should generate a code number for this medicine which the shop owner would paste the code number in the rack where this medicine would be stored. The shop owner should be able to query about a medicine either using its generic name or the trade name and the software should display its code number and the quantity present.
- At the end of every day the shop owner would give a command to generate the list of medicines which have expired. It should also prepare a vendor-wise list of the expired items so that the shop owner can ask the vendor to replace these items. Currently, this activity alone takes a tremendous amount of labour on the part of the shop owner and is a major motivator for the automation endeavour.
- Whenever any sales occurs, the shop owner would enter the code number of each medicine and the corresponding quantity sold. The MSA should print out the cash receipt.

- The computer should also generate the revenue and profit for any given period. It should also show vendor-wise payments for the period.

### **3. Overall Description**

#### **3.1 Product Perspective**

A retail medicine shop deals with a large number of medicines procured from various manufacturers. For systematic and efficient functioning of the shop, the shop owner needs to maintain his inventory in a well ordered manner. The Medicine Shop Automation Software not only helps in inventory management following JIT philosophy, it also provides a centralized overview of the functioning of the medicine shop. All transactions of the medicine shop can be recorded and the database updated accordingly. The database not only stores information about the medicines, but also required details of the vendors the shop-owner deals with. The system can also generate the list of expired items to be replaced, new items to be ordered and keep track of the revenue and profit in overall or period-wise transactions.

In this way, the MSA provides the much needed automation to reduce labour and increase efficiency in the customer dealings of the medicine shop.

#### **3.2 Product Functions**

The main purpose of the Curiosity MSA is to maximize user utility. It can handle a wide range of user-specific queries.

The MSA allows the shop owner to efficiently handle huge amounts of information regarding a large number of medicines by allowing him/her to perform the following functions:

- Create records of any new medicine procured by the shop
- Update existing records of medicines that are already in stock
- Query the details of any medicine by using either its Trade Name or its Generic Name
- Generate the list of all medicines whose stock needs to be replenished
- Generate the list of all medicines which have expired.

In addition, the Curiosity MSA also automatically performs useful functions such as

- Print out cheques in favour of the vendor when medicines are purchased.



- Print out the cash receipt every time the shop sells a medicine.
- Calculate the threshold value of every medicine and ensure that every medicine is in stock.

The Curiosity Medicine Shop Automation should embody the following features and functions:

- Should be able to enforce medicine purchases according to sales performance of each medicine.
- The calculation of sales performance should be dynamic, and not static, which is a necessary implication of the JIT philosophy, so that the quantity of medicines purchased is only the quantity that is necessary.
- Should be scalable, which means that its performance should not degrade with the addition of new medicine records and/or vendor records to the database.
- Should be configurable, and allow handling of other specific queries that may be necessary for required by the shop-owner.
- Should provide administrative security in terms of a password-protected login system for the shop-owner.

### **3.3 User Characteristics**

The targeted user for the MSA is the owner of the medicine retail shop. The MSA can be operated even by a layman possessing basic knowledge about handling software. Therefore, no specialized skills are needed from the user to use the interface.

### **3.4 Constraints**

The probable constraint on the project is the availability of a proper computing platform setup in the medicine retail shop. The software requires a computer with no specialized features other than the minimum additional hardware like a printer to implement the functionality of printing cheques and cash-memos. Its overall functionalities are platform-independent. However, since the software is coded in JAVA with MySQL for handling large data, the computer needs to be pre-installed with dependencies like an updated JVM to run the software. Moreover, a working internet connection will also be needed for proper update and maintenance of the database.

For testing purposes, any computer with pre-installed JVM and access to the

Internet can be used.

### **3.5 Assumptions and dependencies**

A number of factors that may affect the requirements specified in the SRS include:

#### **ASSUMPTIONS:**

- The database records for the medicines which the shop owner deals in are created at the time of procurement of new medicines which the shop-owner had never sold before. The records are updated every time a fresh stock arrives or some transaction occurs.
- The quantity for the initial stocks for the medicine shop must be decided upon by the shop owner since the calculation of the threshold value for JIT will need the sales figures for at least one week.
- In case the shop stops selling a particular medicine, perhaps because it's sale has been banned or it went out of production, the shop-owner must himself/herself delete the record for that particular medicine from the database.
- In case the shop no longer buys medicines from a certain vendor, the shop-owner must also delete the record corresponding to that particular vendor from the database.

#### **DEPENDENCIES:**

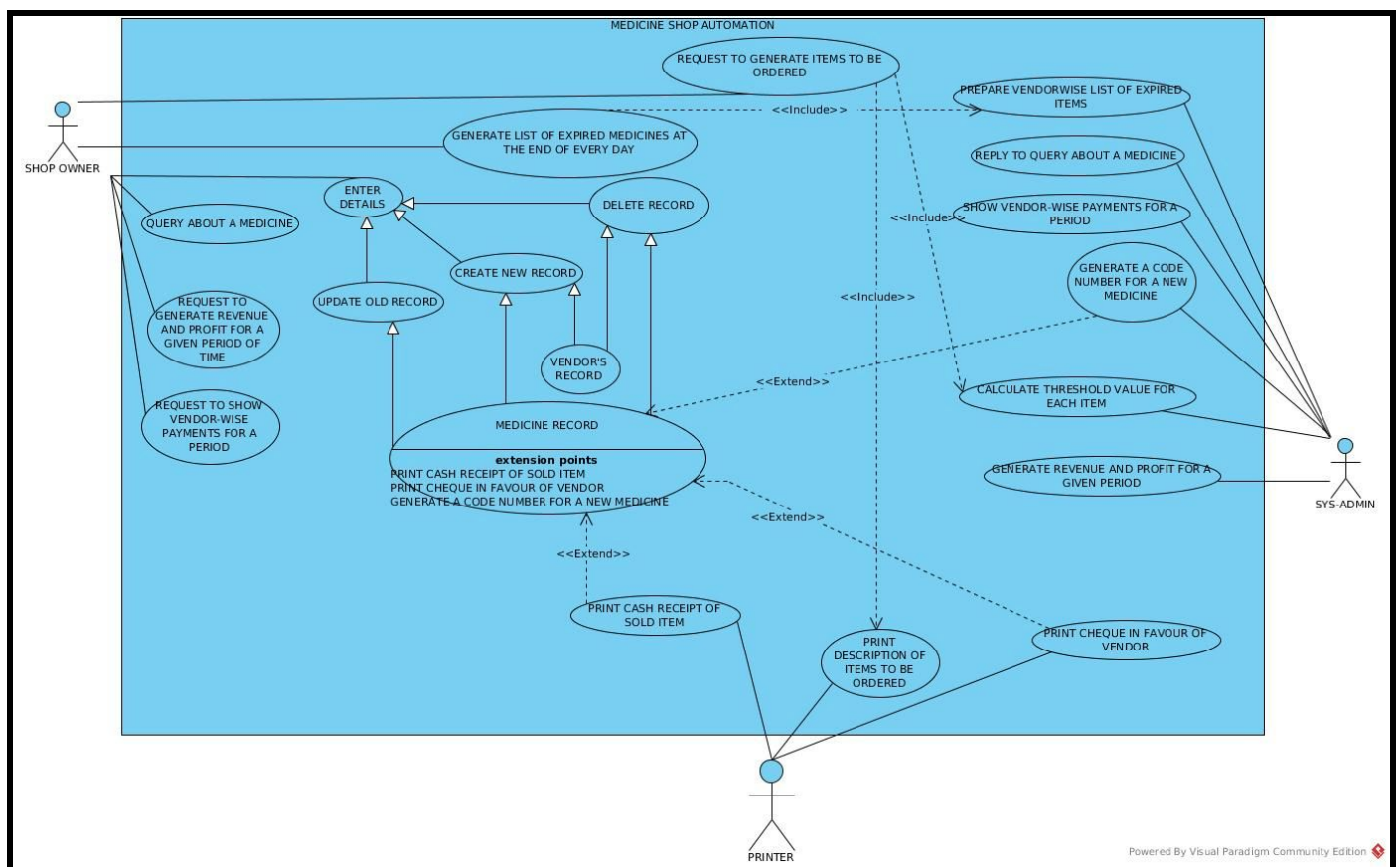
- The Curiosity MSA depends on the status of the MySQL server as the automation relies on the MySQL service to store the required information regarding medicines and the vendors.

### **3.6 Apportionment of requirements**

At present, the Curiosity MSA has been designed with a view of only the shop-owner as the target user. However, there is a scope to extend the MSA service to other shop-employees and the customers in the later versions of the MSA.

## 4. Specific Requirements

### 4.1 Use-case Diagram and Documentation



### DOCUMENTATION:

#### Actors -

- Owner

- System ( Passive Actor )
- Printer ( Secondary Actor )

#### **Use-cases for each Actor -**

- Owner
  - Request to generate items to be ordered
  - Generate list of expired medicines
  - Enter details to:
    - Create a new record
    - Delete a record
    - Update an already existing record

The record can be either a medicine record or a vendor's record.
  - Query about a medicine
  - Request to generate revenue and profit for a certain period of time
  - Request to show vendor-wise payments for a certain period
- System
  - Prepare vendor-wise list of expired items
  - Reply to query about a medicine
  - Generate a code number for a new medicine
  - Show vendor-wise payments for a period
  - Calculate threshold value for each item
  - Generate revenue and profit for a certain queried period of time
- Printer
  - Print cheque in favour of vendor
  - Print description of items to be ordered
  - Print cash receipt of sold item

#### **Brief Description of the Use Cases -**

##### **Use Cases pertaining to interactions with the Shop Owner:**

- **REQUEST TO GENERATE ITEMS TO BE ORDERED:**

At the end of each day's sale, the shop-owner directs the MSA to generate the items to be ordered so that the shop has enough stock to sustain a week's sale following the JIT philosophy.

- ❖ TRIGGER : The Shop-owner selects the option to generate the list of items to be ordered.
  - ❖ PRE-CONDITION : The Shop-owner is logged in to the system. It is the end of the day's transactions at the shop and all the day's transactions have been recorded.
  - ❖ BASIC PATH :
    - The Shop-owner selects the option to generate the list of items to be ordered.
    - The threshold value of each medicine is computed using the average sales figures over the past four weeks via the *includes* relationship with CALCULATE THRESHOLD VALUE FOR EACH ITEM use case of the System-Admin.
    - The threshold value is compared with the quantity of current stock for each medicine.
    - The entire list of medicines to be ordered is generated.
    - The Computer directs the printer to print the list of medicines via the *extends* relationship with the PRINT DESCRIPTION OF ITEMS TO BE ORDERED use case of the Printer.
  - ❖ ALTERNATE PATH : None
  - ❖ POST-CONDITION : The printed list of medicines to be ordered is obtained.
  - ❖ EXCEPTION PATH : In case when the sales figures are not available for the required period, the use case will be abandoned.
- GENERATE LIST OF EXPIRED ITEMS: The shop owner can direct the MSA to generate the list of expired medicines along with the details of the vendor from whom they were bought so that the expired items can be replaced.
    - ❖ TRIGGER : The shop-owner selects the option to generate the list of expired items.
    - ❖ PRE-CONDITION : The shop-owner is logged in to the system.
    - ❖ BASIC PATH :
      - The shop-owner selects the option to generate the list of expired items.
      - The vendor-wise list of expired medicines is displayed via the *includes* relationship with the PREPARE VENDOR-WISE LIST OF EXPIRED ITEMS use case of the System-Admin.

- The expired medicines are sent to the corresponding suppliers to be replaced by fresh ones.
  - ❖ ALTERNATE PATH : None
  - ❖ POST-CONDITION : The expired medicines are returned to the suppliers for replacement using the generated vendor-wise list of expired medicines.
  - ❖ EXCEPTION PATH : None
  - ❖ OTHER : This use case may have an *includes* relationship with the use case PRINT DESCRIPTION OF ITEMS TO BE ORDERED of the Printer. Then the generated list will be printed also for the convenience of the shop-owner.
- ENTER DETAILS: The details pertaining to either a medicine or some vendor the shop-owner deals with, have to be entered into the MSA. It forms a base use case for three other use cases:
  - ➔ CREATE A NEW RECORD : Add a new medicine record or vendor's record to the database.
    - TRIGGER : The shop-owner selects the option to create a new record.
    - PRE-CONDITION : The shop-owner is logged in to the system.
    - BASIC PATH :
      - The shop-owner selects the option to create a new record.
      - The shop-owner is presented with the option to select record type- Medicine record or Vendor's record.
      - In case of Medicine record, a unique code number is generated for the new medicine being recorded via the *includes* relationship with the GENERATE A CODE NUMBER FOR A NEW MEDICINE RECORD use case of the System-Admin. The shop-owner enters the medicine name (trade name as well as generic name), description and code number for the medicine to create the new record.
      - In case of Vendor's record, the shop-owner enters the name, address of the vendor and the pre-assigned Vendor id to create a new record. A list of medicines which the vendor can supply must also be entered to be stored in association with the vendor's record.
      - The new record is entered into the database.
    - ALTERNATE PATH : None
    - POST-CONDITION : The database is now updated with one more

record of a vendor / medicine.

- EXCEPTION PATH : One or more of the necessary details if not provided, the use case will be abandoned.

→ DELETE A RECORD : Delete a medicine or vendor's record to the database.

- TRIGGER : The shop-owner selects the option to delete a record.
- PRE-CONDITION : The shop-owner is logged in to the system. There are few records already in the database.
- BASIC PATH :
  - The shop-owner selects the option to delete a record.
  - The shop-owner is presented with the option to select record type- Medicine record or Vendor's record.
  - The shop-owner is asked to provide the record identification token- code number for the medicine record or vendor id for the vendor's record.
  - The Computer looks up the corresponding record in the database.
  - The record when found is deleted.
- ALTERNATE PATH : None
- POST-CONDITION : The database is now updated with a medicine / vendor's record deleted.
- EXCEPTION PATH : The use case is abandoned if no record exists with the corresponding type and identification token provided by the shop-owner.

→ UPDATE AN ALREADY EXISTING RECORD : Update the stock and other details pertaining to a medicine record when new stock arrives or when some medicines are sold.

- TRIGGER : The shop-owner selects the option to update record when new stock arrives or when medicines are sold.
- PRE-CONDITION : A fresh stock of medicines have been supplied or some medicines are sold and the corresponding details must be accordingly tabulated. The shop-owner is logged in to the system.
- BASIC PATH :
  - The shop-owner selects the option to update record when new stock arrives or when medicines are sold.

- In case of stock arrival, the shop-owner enters the batch number, expiry date, unit purchasing price, quantity of new stock, date of arrival and the Vendor id of the supplier. The quantity of current stock for that medicine is correspondingly updated. The cheque in favour of the vendor is printed via the *extends* relationship with the PRINT CHEQUE IN FAVOUR OF THE VENDOR use case of the Printer.
  - In case of a medicine being sold, the shop-owner enters the batch number, expiry date, per unit selling price, quantity sold and date of sale. The quantity of current stock for that medicine is correspondingly updated. The cash receipt for the customer is printed via the *extends* relationship with the PRINT CASH RECEIPT OF SOLD ITEM use case of the Printer.
  - ALTERNATE PATH : None
  - POST-CONDITION : The database is now updated with the medicine record.
  - EXCEPTION PATH : The use case is abandoned if there are interruptions in the update procedure either due to insufficient stock or unaccepted standards in the transactions.
- QUERY ABOUT A MEDICINE : The owner can use the MSA to query about a medicine using its trade name or generic name.
    - TRIGGER : The shop-owner selects the option to search for a medicine in the database of medicine records.
    - PRE-CONDITION : The shop-owner is logged in to the system. There are few records already present in the database.
    - BASIC PATH :
      - ➔ The shop-owner selects the option to search for a medicine in the database of medicine records.
      - ➔ The shop-owner is given the option to search either by trade name or generic name.
      - ➔ The System-Admin looks up the medicine record, if exists, in the database of medicine records via the REPLY TO QUERY ABOUT A MEDICINE use case of the System-Admin.
      - ➔ If the required record is found in the database, the corresponding medicine details are displayed.



- ALTERNATE PATH : None
  - POST-CONDITION : The record of the medicine with corresponding trade name or generic name is displayed.
  - EXCEPTION PATH : The use case is abandoned if either no record exists in the database yet or no record exists with corresponding trade name or generic name, as required by the shop-owner.
- 
- REQUEST TO GENERATE REVENUE AND PROFIT FOR A CERTAIN PERIOD OF TIME : The owner can direct the MSA to generate revenue and profit for the sales of a certain period of time.
    - TRIGGER : The shop-owner selects the option to generate revenue and profit figures for the sales.
    - PRE-CONDITION : The shop-owner is logged in to the system.
    - BASIC PATH :
      - The shop-owner selects the option to generate revenue and profit figures for the sales.
      - He/She is prompted to enter the time period for which the figures must be generated.
      - The shop-owner is also prompted to provide the rate of taxation for calculating the revenue.
      - The revenue and profit figures are generated via the GENERATE REVENUE AND PROFIT FOR A CERTAIN QUERIED PERIOD OF TIME use case of the System-Admin.
      - The results are displayed to the shop-owner.
    - ALTERNATE PATH : None
    - POST-CONDITION : The shop-owner obtains the required data regarding revenue and profit from sales for the period of time he/she specifies.
    - EXCEPTION PATH : The use case is abandoned if the sales records are not available for a part or whole of the period specified.
    - OTHER : The use case could also include the functionality of plotting these figures over time for better visual presentation to the shop-owner.
  
  - REQUEST TO SHOW VENDOR-WISE PAYMENTS FOR A CERTAIN PERIOD OF TIME : The owner can direct the MSA to show vendor-wise payments for a definite period of time as and when required.

- TRIGGER : The shop-owner selects the option to show the vendor-wise payments made by him.
- PRE-CONDITION : The shop-owner is logged in to the system.
- BASIC PATH :
  - The shop-owner selects the option to show the vendor-wise payments made by him.
  - He/She is prompted to enter the time period for which the figures must be computed.
  - The figures for vendor-wise payment for the specified period is generated via the SHOW VENDOR-WISE PAYMENTS FOR A CERTAIN PERIOD OF TIME use case of the System-Admin.
  - The results are displayed to the shop-owner.
- ALTERNATE PATH : None
- POST-CONDITION : The shop-owner obtains the required data regarding vendor-wise payments for the period of time he/she specifies.
- EXCEPTION PATH : The use case is abandoned if payment records are not available for a part or whole of the period specified and for all vendors the shop-owner deals with.
- OTHER : The use case could also include the functionality of plotting these figures vendor-wise over time for better visual presentation to the shop-owner.

#### **Use Cases Pertaining to the System-Admin :**

- PREPARE VENDOR-WISE LIST OF EXPIRED ITEMS : The system is capable of scanning the database to prepare the vendor-wise list of supplied medicines that have expired as and when requested by the shop-owner.
  - ❖ TRIGGER : The Shop-owner selects the option to generate the list of medicines that have expired.
  - ❖ PRE-CONDITION : The Shop-owner is logged in to the system. It is the end of the day's transactions at the shop and all the day's transactions have been recorded.
  - ❖ BASIC PATH :
    - The Shop-owner selects the option to generate the list of medicines that have expired using the use-case GENERATE LIST OF EXPIRED ITEMS AT THE END

OF EACH DAY.

- The expiry date of each medicine is compared with the current date.
  - The entire vendor-wise list of medicines that have expired is generated.
  - ❖ ALTERNATE PATH : None
  - ❖ POST-CONDITION : The vendor-wise list of medicines that have expired is sent to the Shop-owner.
  - ❖ EXCEPTION PATH : None.
- REPLY TO QUERY ABOUT A MEDICINE : The MSA system is capable of fetching data from the database to provide the queried details to the user.
    - ❖ TRIGGER : The Shop-owner searches for a particular medicine using its Trade Name or its Generic Name.
    - ❖ PRE-CONDITION : The Shop-owner is logged in to the system.
    - ❖ BASIC PATH :
      - The Shop-owner searches for a medicine using its trade name or generic name using the use-case QUERY ABOUT A MEDICINE.
      - The entered Trade Name / Generic Name is compared with the Trade Name / Generic Name of each medicine
      - The information of all matching cases is generated
    - ❖ ALTERNATE PATH : None
    - ❖ POST-CONDITION : All matching records are sent back to the Shop-owner. In case of no match, an error message is generated.
    - ❖ EXCEPTION PATH : None.
- GENERATE A CODE NUMBER FOR A NEW MEDICINE RECORD : Whenever the user adds a new medicine record to the database, the system generates a unique identification code for that medicine to be referred to in all future customer transactions and vendor dealings.
    - ❖ TRIGGER : The Shop-owner selects the option to create a new record.
    - ❖ PRE-CONDITION : The Shop-owner is logged in to the system.
    - ❖ BASIC PATH :
      - The Shop-owner selects the option of creating a new record under the use-case CREATE NEW RECORD

- The Shop-owner selects the type of record in the use case MEDICINE RECORD
  - A code number is generated on the basis of the Trade Name, Generic Name and the batch number of the medicine record.
  - ❖ ALTERNATE PATH : None
  - ❖ POST-CONDITION : The generated code-number is returned to the Shop-Owner.
  - ❖ EXCEPTION PATH : None.
  
- SHOW VENDOR-WISE PAYMENTS FOR A CERTAIN PERIOD OF TIME : The System can query the database to determine and display the vendor-wise payments made by the shop-owner for a certain period of time as queried by the user.
  - ❖ TRIGGER : The Shop-owner selects the option to show vendor-wise payments.
  - ❖ PRE-CONDITION : The Shop-owner is logged in to the system.
  - ❖ BASIC PATH :
    - The Shop-owner selects the option to show vendor-wise payments.
    - The shop-owner specifies the period for which the payments need to be generated under the use-case REQUEST TO GENERATE VENDOR-WISE PAYMENTS FOR A GIVEN PERIOD OF TIME.
    - The payment of each vendor in the given period is calculated and generated
  - ❖ ALTERNATE PATH : None
  - ❖ POST-CONDITION : The list of vendor-wise payments is returned back to the shop-owner.
  - ❖ EXCEPTION PATH : None.
  
- CALCULATE THRESHOLD VALUE FOR EACH ITEM : In order to prepare the list of items to be ordered, the MSA must calculate the weekly average sales of all items. If the current stock level for any medicine is below the expected sale amount, that medicine must be added to the list of items to be ordered. Thus by calculating the threshold value and current stock of each item, the MSA system maintains the JIT philosophy.
  - ❖ TRIGGER : The Shop-owner selects the option to generate list of

items to be ordered.

- ❖ PRE-CONDITION : The Shop-owner is logged in to the system. It is the end of the day's transactions and all of the day's transactions have been recorded in the system.
- ❖ BASIC PATH :
  - The Shop-owner selects the option to generate the list of items to be ordered with the use case REQUEST TO GENERATE ITEMS TO BE ORDERED.
  - The average of the sales of each item for the last 4 weeks is calculated separately. These values are generated as the threshold value of each item.
- ❖ ALTERNATE PATH : None
- ❖ POST-CONDITION : The Threshold values of each medicine is returned to the shop-owner.
- ❖ EXCEPTION PATH : None.

- GENERATE REVENUE AND PROFIT FOR A CERTAIN QUERIED PERIOD OF TIME : The System can query the database to determine and generate the revenue and profit from sales for a certain period of time as queried by the user.

- ❖ TRIGGER : The Shop-owner selects the option to show revenue and profit.
- ❖ PRE-CONDITION : The Shop-owner is logged in to the system.
- ❖ BASIC PATH :
  - The Shop-owner selects the option to show revenue and profit.
  - The shop-owner specifies the period for which the revenue and profit need to be generated as well as the rate of taxation under the use-case REQUEST TO GENERATE REVENUE AND PROFIT FOR A GIVEN PERIOD OF TIME.
  - The total amount spent by the medicine shop in the given period is calculated.
  - The total value of medicines sold in the given period is calculated.
  - The difference between these two amounts is calculated and is generated as profit.
  - The total revenue is also generated on the basis of the amount spent and the rate of taxation.

- ❖ ALTERNATE PATH : None
- ❖ POST-CONDITION : The revenue and profit is returned to the shop-owner.
- ❖ EXCEPTION PATH : None.

### **Use Cases pertaining to the Printer :**

- **PRINT CHEQUE IN FAVOUR OF THE VENDOR** : Whenever fresh supply of medicines arrives from a certain vendor, the MSA system, on queue from other use cases instructs the printer to print cheque in favour of the vendor to make the required payments.
  - TRIGGER : The shop-owner selects the option to update record when new stock arrives.
  - PRE-CONDITION : A fresh stock of medicines have been supplied and the corresponding details must be accordingly tabulated. The shop-owner is logged in to the system. The printer must also be properly set up along with the computer.
  - BASIC PATH :
    - The shop-owner selects the option to update record when new stock arrives.
    - The shop-owner enters the batch number, expiry date, unit purchasing price, quantity of new stock, date of arrival and the Vendor id of the supplier. The quantity of current stock for that medicine is correspondingly updated.
    - The cheque in favour of the vendor is printed by the Printer.
    - The shop-owner makes the necessary payment to the supplier.
  - ALTERNATE PATH : None
  - POST-CONDITION : The database of medicine records is properly updated. The cheque is instantly printed and can be handed over to the vendor after necessary verification.
  - EXCEPTION PATH : The use case will be abandoned if the Printer is not properly set up to function with the Computer.
- **PRINT DESCRIPTION OF ITEMS TO BE ORDERED** : Once the System-Admin has generated the list of medicines to be ordered, the printer provides a printed description of the items to be ordered, along with the corresponding quantities and other details as required.

- TRIGGER : The shop-owner selects the option to generate the list of items to be ordered.
  - PRE-CONDITION : The shop-owner is logged in to the system.
  - BASIC PATH :
    - The Shop-owner selects the option to generate the list of items to be ordered.
    - The threshold value of each medicine is computed using the average sales figures over the past four weeks via the CALCULATE THRESHOLD VALUE FOR EACH ITEM use case of the System-Admin.
    - The threshold value is compared with the quantity of current stock for each medicine.
    - The entire list of medicines to be ordered is generated.
    - The list of medicines is printed by the Printer.
    - The order for medicine is placed as necessary.
  - ALTERNATE PATH : None
  - POST-CONDITION : The shop-owner obtains the printed list of medicines to be ordered, in accordance with the JIT philosophy of inventory management.
  - EXCEPTION PATH : The use case will be abandoned if the Printer is not properly set up to function with the Computer.
- PRINT CASH RECEIPT OF SOLD ITEM : When a medicine is sold, the MSA, following the action of other use cases, directs the printer to provide a printed cash receipt of the sold item with other details and descriptions as decided beforehand.
    - ➔ TRIGGER :
    - ➔ PRE-CONDITION :
    - ➔ BASIC PATH :
      - ◆ The shop-owner selects the option to update record when medicines are sold.
      - ◆ The shop-owner enters the batch number, expiry date, per unit selling price, quantity sold and date of sale. The quantity of current stock for that medicine is correspondingly updated.
      - ◆ The cash receipt for the customer is printed by the Printer.
      - ◆ It is handed over to the customer after the necessary payment is made.
    - ➔ ALTERNATE PATH : None
    - ➔ POST-CONDITION : The shop-owner obtains the printed

cash-receipt of the transaction which he/she can hand over to the customer in exchange of the necessary payment.

→ **EXCEPTION PATH** : The use case will be abandoned if the Printer is not properly set up to function with the Computer.

## **RELATIONS AMONG THE USE CASES :**

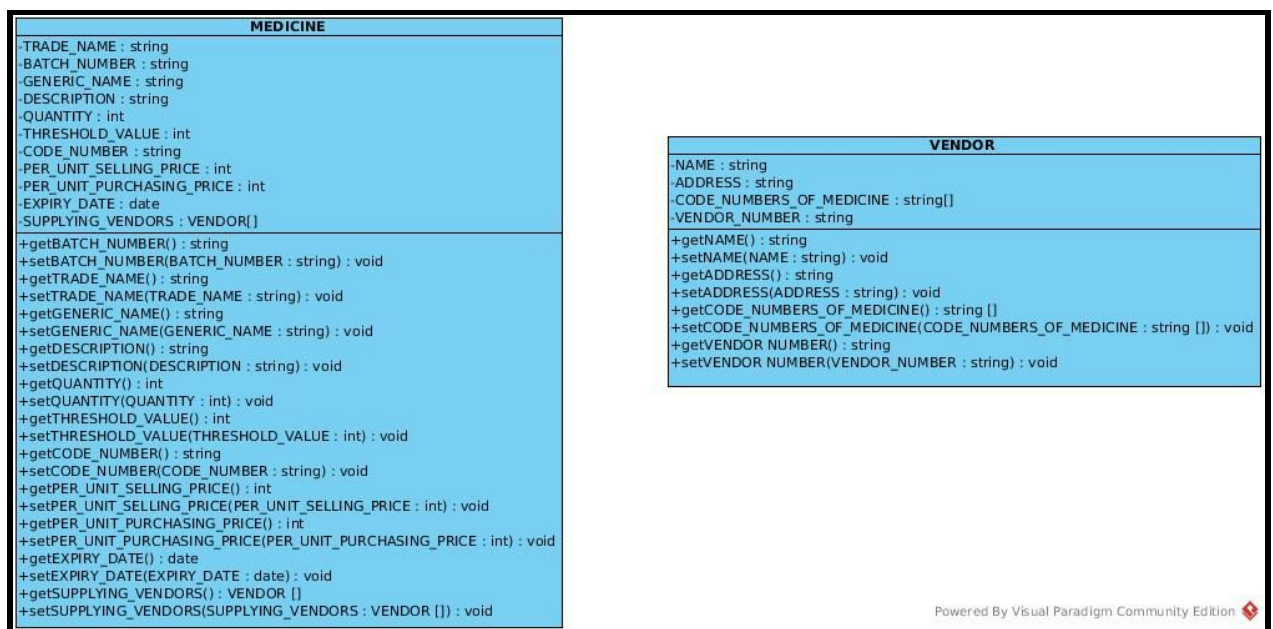
- “Request to generate items to be ordered” use case of the Shop-Owner **includes** the “Calculate threshold value for each item” use case of the System-Admin and also **includes** the “Print description of items to be ordered” use case of the Printer since the latter two use cases complete the request initiated by the shop-owner.
- “Generate list of expired medicines at the end of day” use case of the shop-owner **includes** the “Prepare vendor-wise list of expired items” use case of the System-Admin since the former depends on the latter for its successful execution.
- **Generalization relationship** between the following use cases:
  - “Enter details” and “Create new record”
  - “Enter details” and “Delete existing record”
  - “Enter details” and “Update existing record”

This is because the shop-owner can enter details pertaining to either medicines or vendors to create, delete or update records.
  - “Create new record”, “Delete existing record”, “Update existing record” and “Vendor’s record”
  - “Create new record”, “Delete existing record”, “Update existing record” and “Medicine record”

This is because these actions can be applied to either a vendor’s record or a medicine record in the database.
- “Generate a code number for a new medicine” use case of the System-Admin, “Print cash receipt for sold item” use case of the Printer and “Print cheque in favour of the vendor” use case of the Printer **extends** the “Medicine Record” since either of these use cases will be initiated depending on whether a medicine record is created or updated either due to arrival of new stock or sale from current stock.



## 4.2 Class Diagram and Documentation



### DOCUMENTATION:

#### **Classes -**

- Medicine
- Vendor

#### **Attributes in each Class -**

- Medicine
  - Trade\_Name : string
  - Batch\_Number : string

- Generic\_Name : string
- Description : string
- Quantity : int
- Threshold\_Value : int
- Code\_Number : string
- Per\_Unit\_Selling\_Price : int
- Per\_Unit\_Purchasing\_Price : int
- Expiry\_Date : date
- Supplying\_Vendors : Vendor[]

➤ Vendor

- Name : string
- Address : string
- Code\_Numbers\_of\_Medicine : string[]
- Vendor\_Number : string

**Operations in each Class -**

➤ Medicine

- getBatch\_Number() : string
- setBatch\_Number(Batch\_Number : string) : void
- getTrade\_Name() : string
- setTrade\_Name(Trade\_Name : string ) : void
- getGeneric\_Name() : string
- setGeneric\_Name(Generic\_Name : string) : void
- getDescription() : string
- setDescription(Description : string) : void
- getQuantity() : int
- setQuantity(Quantity : int) : void
- getThreshold\_Value() : int
- setThreshold\_Value() : Threshold\_Value : int
- getCode\_Number() : string
- setCode\_Number(Code\_Number : string) : void
- getPer\_Unit\_Selling\_Price() : int
- setPer\_Unit\_Selling\_Price(Per\_Unit\_Selling\_Price : int) : void
- getPer\_Unit\_Purchasing\_Price () : int
- setPer\_Unit\_Purchasing\_Price (Per\_Unit\_Purchasing\_Price : int) : void
- getExpiry\_Date() : date
- setExpiry\_Date(Expiry\_Date : date) : void
- getSupplying\_Vendors() : Vendors[]
- setSupplying\_Vendors(Supplying\_Vendors : Vendors[]) : void

➤ Vendor

- getName() : string
- setName(Name : string) : void
- getAddress() : string
- setAddress(Address : string) : void
- getVendor\_Number() : string
- setVendor\_Number(Vendor\_Number : string) : void
- getCode\_Numbers\_Of\_Medicine() : string[]
- setCode\_Numbers\_Of\_Medicine(Code\_Numbers\_Of\_Medicine : string[]) : void

Therefore, in the first stage of abstraction of the system according to the user-given document, the class diagram was modelled to represent the given information in terms of two distinct classes : Medicine and Vendor. The attributes were scanned from the document and only getter and setter operations were added to access the attributes. Further, no association, aggregation or composition relationship was analysed between classes.

### **4.3 Purchased Components**

The entire project of developing the Medicine Shop Automation software is open source, and hence no component of it will be purchased.

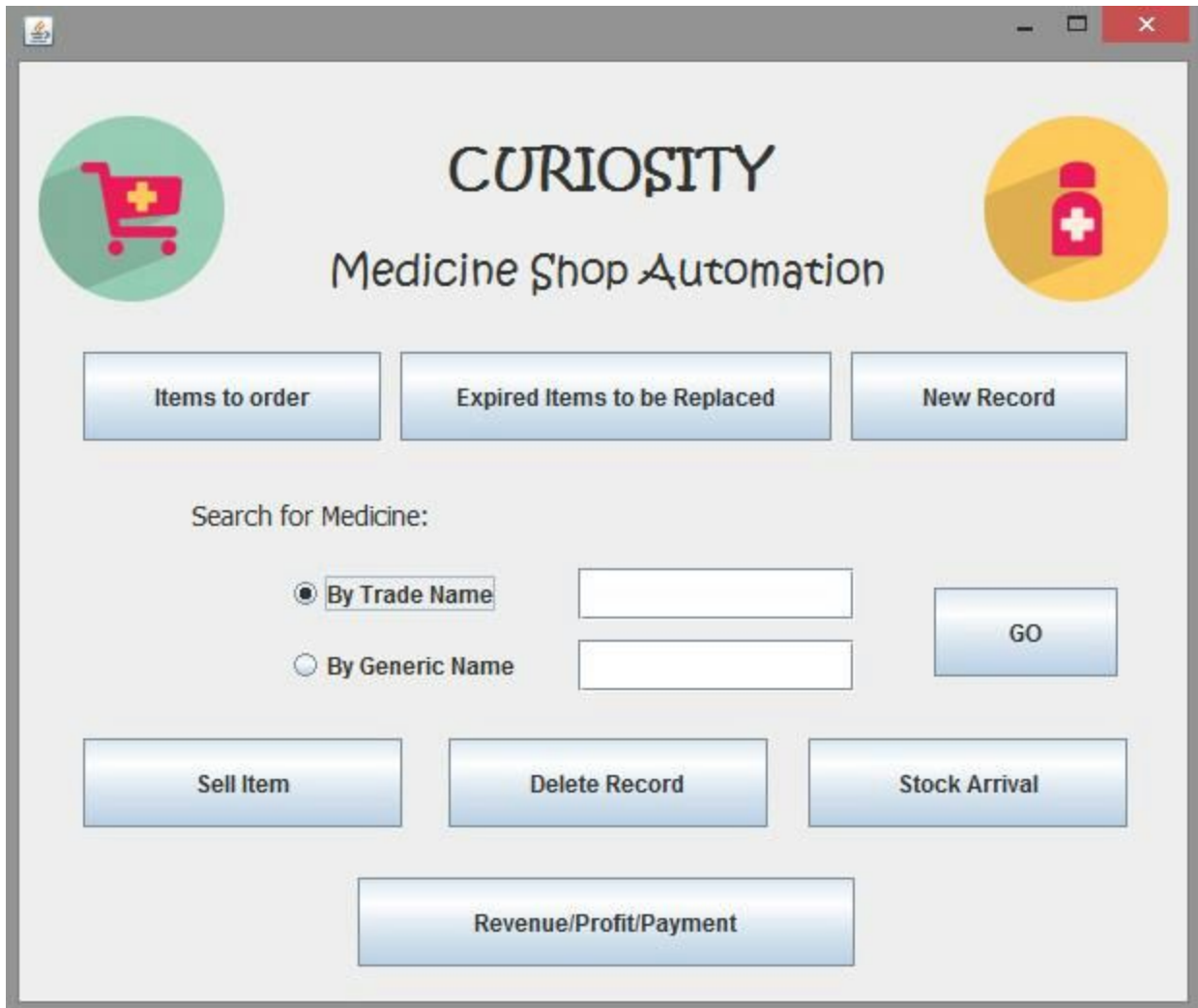
## 4.4 Interfaces

### 4.4.1. LOG IN PAGE



A screenshot of a 'User Login' window. The window has a standard Windows-style title bar with a minimize button, a maximize button, and a close button (red with a white 'X'). The main content area has a light gray background. On the left side, there is a graphic of a blue padlock and a yellow key. To the right of this graphic, the text 'User Login' is centered. Below the title, there are two input fields: 'Username:' followed by a white text box, and 'Password:' followed by a white text box. At the bottom center, there is a blue button with the text 'LOGIN' in white capital letters.

#### 4.4.2. WELCOME PAGE



The screenshot displays the 'CURIO\$ITY Medicine Shop Automation' interface. At the top, the title 'CURIO\$ITY' is in a large, stylized font, with 'Medicine Shop Automation' below it. The interface is flanked by two circular icons: a green one with a red shopping cart and a yellow one with a red pill bottle. Below the title, there are three buttons: 'Items to order', 'Expired Items to be Replaced', and 'New Record'. A search section follows, labeled 'Search for Medicine:', with two radio buttons: 'By Trade Name' (selected) and 'By Generic Name'. Each radio button is followed by a text input field. To the right of these fields is a 'GO' button. At the bottom, there are four buttons: 'Sell Item', 'Delete Record', 'Stock Arrival', and 'Revenue/Profit/Payment'.

**CURIO\$ITY**  
Medicine Shop Automation

Items to order    Expired Items to be Replaced    New Record

Search for Medicine:

☒ By Trade Name

☐ By Generic Name

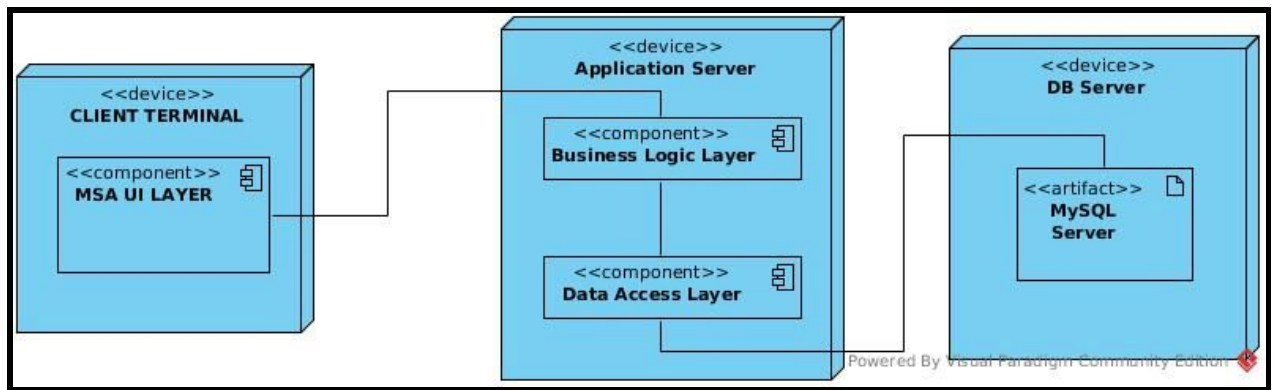
GO

Sell Item    Delete Record    Stock Arrival

Revenue/Profit/Payment

## 4.5 Deployment Diagram :

The following is the Deployment Diagram for the MSA :



The design follows a 3-tier architecture, rather than a 2-tier one. The client layer contains the UI part of the application. In the business layer all business logic like validation of data, calculations, data insertion etc are written. This acts as a interface between Client layer and Data Access Layer. This layer is also called the intermediary layer and helps to make communication faster between client and data layer. Data Access Layer contains methods to connect with database and to perform insert, update, delete, get data from database based on our input data.

The 3-tier architecture was chosen mainly because it provides high degree of flexibility in deployment platform and configuration, better re-use, improved data integrity and improved security as compared to the 2-tier architecture.

## 4.6 Licensing Requirements

The Curiosity MSA will be released under a GPL license and will be completely open-source.