VIRTUALSTORE (VRS)

Souhail Madani

Supervised by:

Dr. Jorge Martinez Ladron de Guevera

**Document Structure:**

1. Introduction
2. Initial Version of User and System Requirements
3. Intermediate Version of User and System Requirements
4. Final Version of User and System Requirements
5. Context Diagram and Conceptual Process Models

5-1 Context Diagram

5-2 Context Diagram Description

5-3 Conceptual Process Models

5-3-1 CPM: Manage Clients

5-3-2 CPM: Manage Providers

5-3-3 CPM: Manage Transactions

5-3-4 CPM: Manage Selling Process

5-3-5 CPM: Manage Warehouse and Factory Communication

5-3-6 CPM: Manage Official Documents

1. Conceptual Architecture

6-1 Diagram

6-2 Description

1. Conceptual Data Models and Entity Relations

7-1 Description of the meeting with the client

7-2 Presentation of the actual forms

7-3 Data Dictionary

7-4 Analysis of Data Dictionary: Identifying Anomalies

7-5 Demonstration of the Completeness of Data

7-6 Description and Justification of the Use of Entities

7-7 Entity Relations Diagram

7-7-1 Description of the Entity Relation Diagram

7-8 Confrontation of Data Models with Process Models

1. Conclusion
2. **Introduction:**

**The User and System Requirements** document is an essential step in **Requirements Engineering** that comes just after the feasibility study. The major goal of this document is to provide a clear and organized idea about the user requirements and specifications that are expressed in a natural language by translating them into technical language. Usually, the client’s requirements are expressed in a general, vague, inconsistent, and ambiguous way. Therefore, we need to go through the requirements engineering process to make these requirements actions ready for automations. This process reflects an iterative model that has four steps, namely **Elicitation**, **Analysis**, **Specification**, and **Validation**. First, there is Elicitation where we should get information from the user. It is about what the client has said. Second, there is Analysis where we develop system models that should be used to study those requirements. This step is about what we think of what the client has said. Third, the specification step dissects and decomposes the information gathered from the users. Finally, the validation part double checks what has been established in the previous sections. This process is iterative, meaning that we keep looping until we make sure that we have a good understanding of what the actual requirements are. In this part, one needs to heavily interact with the user to get as much information as possible. Conducting interviews, using questionnaires are examples of the methods used to extract information about the software. Requirements Engineering involves a process of iteration where requirements are dissected into more detailed ones and so on until we reach technical models that are easy to implement. This is the part of modeling, filtering, and analyzing. During the process of Engineering Requirements, three fundamental elements should be depicted, namely **Users**, **Data**, and **Processes**. The ultimate purpose, as put by **Dr. Martinez Ladron de Guavera**, is to consider these components in light of three essential attributes throughout the whole process, namely **Completeness**, **Precision**, and **Consistency**. Completeness states that our requirements should be comprehensive and including all elements. Precision makes our requirements correct and accurate I n describing the actual specifications. Finally, our requirements should be consistent in the sense that they should not be ambiguous or contradictory. Moreover, throughout the whole process, requirements are to be classified into three main categories, which are functional requirements, nonfunctional requirements, and domain requirements. Actually, the three elements and the three attributes could be gathered together in a matrix.. The next step is to analyte data, processes, and users and their relations. The role of this part is to provide us with a deep understanding of the components of the system. Also, it is going to depict the different buildings blocks of the system along with the respective relationships between them. The **Context Diagram** provides a global perspective of the different users, processes, and data stores that are common to two or more elements in the system. The **conceptual architecture** is to be developed in a simple and clear way, which will show the hierarchy of the processes. This architecture model helps us to have a more practical and clear approach to the hierarchy of processes. The next step is the Conceptual Data Modeling, After meeting the client and getting all the forms needed to extract the data of our software, we will be analyzing those forms to extract **attributes**. This forms and attributes will help us build the **data dictionary**. From this form we will be extracting the **entities**. This is the final sub-step of the **requirement engineering step** before the design step.

1. **Initial Version of User and System Requirements:**

Five Distribution is a medium size company located in Casablanca. It started in business in 2004. Five Distribution has a long-term plan which is launching 50 modern butcher shops. These shops will be selling the products manufactured in the factory. The implementation of the plan already started with launching the first one in Casablanca. Mrs. Mounacir is responsible for monitoring this plan and implementing it. She affirmed that as the number of transactions in the shop is increasing, she finds that managing the shop becomes more difficult. And because the company is willing to launch 49 additional stores, Mrs. Mounacir is convinced that an information system is essential to achieve the plan set by the company. During our first meeting, Mrs. Mounacir and us had a very interesting discussion in which we tried to build a rough view about the system functioning. Following are the questions that we asked our client along with her answers:

* Do you have any idea about the software you need us to implement for your business?

‘‘

What I need in the system, is the ability to follow up all transactions in that happens in the store. I want to keep track of what is bought and what is sold in an efficient manner. I want the follow-up to be quick as we are expecting that the transactions will strongly increase in the future. No need to insist on the security of the software because I do not want a stranger to have access to my store’s information. Transactions should be stored with all their relevant information. I want to be able to check them whenever I want. They must be listed in a clean and clear way.

I want the system to know my customers and store their information. I want to be able to see my history of transactions with him. I need to be able to modify this information. The software should also have a calculator that can generate the total price that the customer should pay.

Another important point is handling the products. I want the system to have all information about the products stored. I want to be able to add a new product to the list and modify its price anytime

The system should be robust. It should update the store after each transaction. Moreover, the system should remind me if there is a shortage in a specific product in the store. Ideally, this shortage warning should be immediately sent to the factory so that they supply us with the appropriate amount of these product.

Suppliers should also be considered in this system. I need to have an archive of the suppliers and my transactions with them. All information concerning these two should be stored safely and efficiently. The store should be updated after each transaction.

I would be glad if the system can automate the invoice and bills writing.

”

* Can you tell us how your stock system works?

“

As in any stock, the flow of products is happening on a daily basis. Every day we are selling large quantities of products and we may receive supplies from our factory. However, products differ in terms of date of expiration, frequency of consumption, and price. Therefore, our main goal is to ensure availability of products. Moreover, we are constantly maintaining a reserve stock in case of running out of products. Simultaneously, we are providing dozens of snacks with delicatessen products and sauces. The store assistant is responsible for keeping track of the inventory and handling the financial aspect of each transactions. An inventory is performed each week to control things.

”

* Which part of your system you want to automate?

“

I want to automate the financial management aspect of the business, which consists of generating ratios and indices of how well we are doing. From the daily inventory to the monthly profit, it would be preferable to generate them using the computer so I can easily manipulate them and archive them if necessary. Automating the management of the stock and its product is as important as automating the financial side. I don’t want to keep checking the shortage on each specific product in the stock, I want to automate this operation. I would like also to generate the bills and reports that I have in my computer.

”

* What are your current computer system specifications?

“

We have an i5 with 8G RAM running under Windows 10 with a hard disk of 200GB. My computer literacy is limited to basic operations or repetitive tasks. I can get use to new software with the aid of tutorials and a certain training period.

”

* After this meeting, the team members decided to meet for a four hours discussion to assess the information gathered, analyze it, and determine the remaining information that we need to know so that we can tackle it in the next meeting. The analysis of the information gathered so far was done through the magic matrix.

|  |  |  |  |
| --- | --- | --- | --- |
| Elements  Attributes | **Data** | **Users** | **Processes** |
| **Completeness** | 70% | 30% | 60% |
| **Consistency** | 40% | 30% | 50% |
| **Precision** | 50% | 30% | 30% |

Table 1: Initial Version Matrix

* **This magic matrix** reflects our understanding of the client’s requirements. For the first component **Data**, after analyzing the information gathered in the first meeting, we could determine the data processed in the shop. It includes products, transaction, customers’ information, and sellers’ information. However**, these are not the only** data used within the shop, which justifies the percentage assigned to **completeness**. Also, we do not have a **clear idea of the relations** between these data as well as **a precise definition of them** which justifies the percentage assigned to **these two criteria** (consistency & precision). For the second component **Users**, we do not know if the system will be used **only by Mrs. Mounacir, or the shop assistants** will also have access to it. We do not know how the system **should be customized to each of the potential users** and the **restrictions on access to each of them**. Therefore, the percentages presented in this column **reflect our poor knowledge about this component**. Concerning the last component **Processes**, we cannot deny that we discussed some of them, but we are not **sure that they are the only ones.** Furthermore, there **some points unclear** in the processes that mainly concern **their functioning and how they are related**. As a result, the percentages assigned in this column appear to be fair and reasonable.

After **developing this magic matrix**, it appears that **a second meeting with our client is primordial**. Therefore, we formulated several questions that will tackle the information that we still need. The next part is the report on the second meeting with Mrs. Mounacir and our interaction with her.

1. **Intermediate version of User and System Requirements:**

In order to elicit the information gathered during the first meeting with our client and to have a more consistent ideas about the requirements, we decided to have another meeting with our client. For this meeting, we prepared in advance some questions related to the vagueness of what the client expressed during our last meeting.

**Questions:**

In this part we will be using the magic matrix to help elicit the information the client provided us to determine the main components of the information system we will be developing. The magic matrix will also help get complete, precise, and consistent specifications.

1. Can you tell us how is it difficult managing the store using the current system?

* This question has as purpose to understand what are the difficulties that the client is facing in the present while managing his store. The objective is to better understand what are the problem that the system is supposed to solve.
  + Client answer: Mrs Mounacir: “ The main software that I am using is Microsoft Excel. However, this system is not adapted to my environment of work, and I have to adjust it to my specific context. In other words, . I have to organize the spreadsheet in a proper format and assign computational functions to some cells. Eventually, I find myself doing the same task repetitively. Other issues that I am facing are: corruption of data, the bad and unusable interface of Excel especially when you are working in a table of more than 50 rows due to the variety of products we are selling. Another problem is The customers’ information search is hard and takes time. The shop manager needs to go through a long word file to find a certain customer. My current system does not keep track of the items in the stock; it is done manually using pens and papers, and I want to change that. Basically, the problem is that I need to update all the information (client, suppliers, products, transactions …) manually which is not efficient nor safe”
  + Team Interpretation: The main problem that software should solve is organization, while focusing on abstraction and not mixing unrelated information in one directory.

1. Can you share with us your thoughts on how do you think the developed software will help you achieve your expansions plans

* This question had as purpose to understand the alignment of the software we will develop for the client with respect to his visions/plans for the future.
  + Client Answer: Mrs Mounacir: “The program will help us have a common platform for our multiple stores which will provide a portability and allow sharing of data between multiple stores. Also, the software will help us reduce the time it take us to process client and suppliers transactions implying the increase of profitability and improvement of client-supplier relationship”
  + Team Interpretation: The system has as a purpose the automation and improvement of the regular work of both administrators and employees. We could have implicitly deducted that the system will have multiple users.

1. What can you tell us about the users of this software? (Magic Matrix)

* This question was based of our interpretation during the meeting to answer the one of the main components of the magic matrix which is users.
  + Client answer: “I will be the only user of the system. I want it to be in French because I think I will be more comfortable with that.”
  + Team Interpretation: We could deduct that our software will have only one user. Also, the system should be in French.

1. Can we say that this software can be divided to the following parts: Client management, suppliers management, transactions management, sales management, factory communication ?

* The purpose of this question is to match our perspective of the software with the client demand and need.
  + Client answer: “Yes exactly, and I would also want to be able to use software for manage my official documents”
  + Team Interpretation: We could understand for this question the functionality of the software, in addition to the main part from which it will be composed.

1. What did you mean by “I would be glad if the system can automate the invoice and bills writing”? Does it relate to the functionality of managing official document you mentioned in your last answer?

* The purpose of this question was to understand what the client meant by “I would be glad” and remove the ambiguity on which this function should be optional or a requirement for the software. Also, confirm its alignment with the last functionality mentioned in the answer of the previous question.
  + Client answer: “This functionality is the one I referred to in my last answer, and I want it to be part of the software and not an optional functionality”
  + Team Interpretation: We were able to conclude that management of official documents is also a part of the software.

Based on those answers and using the Magic Matrix we were able to come up with the sets of processes and data that will compose this software. We sent those criteria to the client by email where we had a response in which the client confirmed the validity of our deduction and expressed that this is what he need. Our client needs can be interpreted in the following way:

* Processes:
  + Manage selling process
  + Manage clients
  + Manage providers
  + Manage warehouse and factory
  + Manage official documents
  + Manage transactions
* Data (what is managed):
  + Products (Name, Barre Code, Price, Inventory, etc.)
  + Clients (Name, Address, Contact information, etc.)
  + Providers (Name, Address, Contact information, etc.)
  + Inventory (Associated product, Quantity, storage, etc.)
  + Transactions (Invoice, Purchase order, Delivery form, etc.)
* After the second meeting with Mrs. Mounacir, the team members held a long meeting in which we analysed the information gathered. Then we developed a magic matrix to assess the components of the system to the three criteria: Completeness, Consistency, and Precision.

|  |  |  |  |
| --- | --- | --- | --- |
| Elements  Attributes | **Data** | **Users** | **Processes** |
| **Completeness** | 100% | 100% | 100% |
| **Consistency** | 95% | 100% | 80% |
| **Precision** | 100% | 100% | 90% |

Table 2: Intermediate Version Matrix

* **The magic matrix** above was the magic matrix we developed after our second meeting. For the first component **data**, we **have all the data** that is used in the shop, how **they are classified**, as well as **a precise description of their attributes**. This justifies the percentages assigned in this column. Concerning the second component **Users**, the client gave **us the main user** of the system, as well as additional **information concerning the language it masters**. Therefore, the percentages presented in the corresponding column seem to be reasonable. For the last component **Processes**, the client provided us with **all the processes adopted** in the shop. **Their functioning** was discussed in the first meeting, but **more insights were revealed** in this second meeting. As a result, the percentages in the corresponding column seem to be fair.

As can be seen, **the magic matrix guarantees us that the requirements are complete, consistent, and precise**. The next part is a stable version of the user system requirements.

1. **Final version of User and System Requirements:**

**Section A : Functional Requirements**

1. **Manage selling process bloc :**
   1. Function display available types of products in the store (Delicatessen / Sauces / Meat)
   2. Function display products based on their type
   3. Function select products and add them to the shopping list
   4. Function deselect products and delete them from the shopping list
   5. Function sum total price of the products in the shopping list and save the data in a slot accessible by the managing official documents bloc
2. **Manage clients :**
3. Function add a client to the database
4. Function modify a client information
5. Function view table of clients next to their attributes (Name / Address / Contact information)
6. Function search for a client in the database
7. Function view a specific client attributes
8. Function remove a client from the database
9. **Manage providers :**
10. Function add provider
11. Function delete provider
12. Function modify attributes of provider (Name / Address / Contact information)
13. Function view list of providers
14. Function view a specific provider attributes
15. Function display activity description of the provider
16. Function search provider based on the selected attributes
17. Function sort providers based on their selling products/date/seniority
18. Function display activity description of the provider
19. Function send email to communicate with the client
20. **Manage warehouse and factory communication :**
21. Function send need request for a product (specified product(s) and quantity)
22. Function receive a confirmation of a product delivery
23. Function send request to return an expired product
24. Function send inventory information of the products available in the warehouse (products and their respective quantity remaining in storage)
25. **Manage official documents :**
26. Function generate a PDF formatted invoice for a chosen customer (To be saved in a directory specified by the client)
27. Function generate a PDF formatted purchase order
28. Function generate a PDF formatted delivery form
29. Function scan and store invoice provided by the provider
30. **Manage transaction :**
31. Function add transaction
32. Function modify transaction
33. Function cancel/remove transaction
34. Function sort transaction based on the selected attributes (client / provider / date / amount of the transaction / type of transaction: Purchase or sell)
35. Function search transaction given the selected attributes
36. Function display recent transactions (For an interval of time)

**Section B : Non Functional Requirements**

1. Computer with an i5 with 8G RAM running under Windows 10 with a hard disk of 200GB Windows 7 compatible software
2. Simplified and practical user interface
3. Main language: French
4. **Context Diagram and Conceptual Process Models (CPMs)**

**5-1 Context Diagram**

**A screenshot of a cell phone

Description automatically generatedA screenshot of a cell phone

Description automatically generated**

**5-2 Context Diagram Description**

After completing the step of “User and System Requirement,” the next step is the Context Diagram. In this step we have a global view of the different processes and users that are in the system as well as the archives of data needed by the software. Furthermore, we will visualize clearly the relations between data, users, and processes: we will show the processes that the user will be able to do and the archives of data to which he will need access to them to accomplish these processes. The processes that the system is going to have are: **Manage Clients, Manage Providers, Manage Transactions, Manage Selling Process, Manage Warehouse and Factory Communication, Manage Official Documents**. To complete these processes, the system will need four stores: **Clients Database, Providers Database, Transactions Database, and Products Database**. The system will have only **one user** as affirmed by our client.

**Manage Clients:** this process mainly gathers the different functions of viewing and searching for clients’ information. Moreover, it allows adding a new client to the database, modify a client information in the database, or removing it completely. To accomplish these different functions, the process will need access to the clients’ database as the context diagram shows below.

**Manage Providers:** this process concerns mainly the different functions of viewing and searching for providers’ information. Moreover, it allows adding a new provider to the database, modify a provider’s information in the database, or removing it completely. To accomplish these different functions, the process will need access to the providers’ database as the context diagram shows below.

**Manage Transactions:** This process groups the functions of viewing transactions or searching for a specific one in the archive. In addition, it concerns the necessary functions of adding a transaction to the database, modifying its attribute, or removing it. Therefore, this process will need to be related to the transactions’ database.

**Manage Selling Process:** This process will allow the user to view the products that are in the store and their total price. Besides, it will allow him to add a new product to the shopping list of a sale. An update of the quantities is done whenever a sale is completed. Therefore, this process will need access to the products database.

**Manage Warehouse and Factory Communication:** This process shows how the communication between the factory and the store will be automated. The communication will be done through emails. The main functions under this process are: communicating a need for a product, communicating the need to return an expired product, communicating the inventory information, and receiving delivery confirmation. As a result, the process will need access to the products database.

**Manage Official Documents:** This process shows how the invoices and bonds writing will be automated. The main functions that are under this process are: generating a PDF formatted invoice for a chosen customer, generating a PDF formatted purchase order, generating a PDF formatted delivery form, and storing the invoice provided by a provider.

**5-3 Conceptual Process Models**

**5-3-1 Manage Providers**

A close up of text on a white background

Description automatically generated

Manage Providers process is a process that allows the user to manage the providers of his store. The subprocesses under Manage Providers processes are : add provider, delete provider, modify attributes, view list of providers, view a specific provider, display activity description, search provider, sort providers, and send email. This process is linked to the provider’s database and can only be accessed by the user. The shop has many providers which justifies the necessity of having a database for providers. The subprocesses that under the scope of this process are:

\***Add a Provider:** This subprocess will allow the user of the store to add a provider to the database. The business context of the shop contains different situations in which this process is needed. It will allow the administrator of the system to add a provider to the database of the providers. A notice will pop up confirming whether the provider is successfully added or not.

**\*Delete a Provider:** is a subprocess that will allow the user of the store to delete a provider permanently from the database. A notice will pop up confirming whether the provider is successfully removed or not.

**\*Modify a Provider’s Attributes:** Again, in some situations, the user will need to modify some information related to a provider. This is the role of this subprocess. It allows the user to change the attributes of a provider who is existing in the database.

**\*Search for a Provider:** This subprocess will allow the user to look for a provider by his name. It facilitates the access to the database of providers for the client.

**\*View the List of Providers:** is a subprocess that allows the user to see all his providers on a list that can be either sorted or not.

**\*View a Specific Provider:** is a process that allows the owner to see all the attributes of one of the providers stored in the corresponding database.

**\*View Activity Description:** is a subprocess that displays the recent transaction with this provider.

**\*Sort Providers:** is a subprocess that sorts the providers in the database based on the amount of transactions or names.

**\*Send email:** is a function that allows the user to send an email to the provider to communicate with them about extending the contract or making a request.

**5-3-2 CPM: Manage Clients:**

**A close up of text on a white background

Description automatically generated**

Manage Clients process will help our client to manage his customers. It contains the following sub processes: add a client, modify a client, view table of clients, search for a client, view a specific client, and remove a client. This process is linked the store’s client’s database. As our client will be the only one using the system, he will be the only one able to use this process. Here is below a detailed description of each subprocess:

**\*Add a Client:** This subprocess enables the user to add a client to the database. The business context of our client’s company can have different situation when this process will be useful. For instance, whenever the shop has a new customer, the user will be able to add him to the database. For this reason, this subprocess is linked to the clients’ database. A notice will pop up confirming whether the client is successfully added or not.

**\*Modify a Client:** This subprocess enable the user to change the attributes of its clients stored in the corresponding database. Again, our client may need to modify his customers’ information which makes this subprocess primordial. For the completion of this task, the subprocess will need access to the database of the clients.

**\* View Table of Clients:** This sub-process allows the user to see the list of all the clients in the database in an organized table that is sorted alphabetically in a way that displays the clients in rows and the corresponding attributes in columns.

**\* Search for a Client:** This is a sub-process that allows the user to search for a specific client by entering his last name and ID, in order to review transactions for a refund, warranty service or to modify a client quickly without going through the whole list manually. It is done by reading from the employee database, that is why it is linked to the clients database.

**\*View a Specific Client:** This is a sub-process that allows the user to view a specific client that they have searched for using the search function by clicking on him which will show them a its corresponding attributes.

**\*Delete a Client:** This is a sub-process that will help the user to remove a client from the database of clients. A notice will pop up confirming whether the client is successfully removed or not.

**5-3-3 Manage Transactions**

**A close up of text on a black background

Description automatically generated**

Manage Transactions process allows the user to manage the shop’s transactions. It groups the following sub-processes: add transaction, modify transaction, cancel/remove transaction, sort transaction, search transaction, and display recent transactions. It is linked to the store’s transactions database because all subprocesses need access to it to be completed. The following are the subprocesses of this process:

**\*Add a transaction:** This subprocess allows the user to add a transaction that happened in his shop to the database of transactions. A notice will pop up confirming whether the transaction is successfully added or not.

**\*Modify a Transaction:** This subprocess allows the user to modify a transaction that happened before. The modification mainly concerns the amount of transaction and the provider or the customer with the whom the transaction happened. That is why this subprocess needs access to the transactions database.

**\* Delete a Transaction:** This subprocess allows the user to remove a transaction that already occurred from the database of transactions. A notice will pop up confirming whether the transaction is successfully removed or not.

**\*Sort Transactions:** This subprocess will allow the user to sort the list of transactions by the date of their occurrence or by the amount of money.

**\*Search for a transaction:** This subprocess will allow the user to search for a transaction either by the date of its occurrence, or the name of the provider/customer with whom the transaction took place.

**\*View Transactions for a Period of Time:** This subprocess allows the user to view the list of transactions that happened in his shop. The interval of time is set to a maximum of the recent transactions that he can see is set to 2 months.

**5-3-4 Manage Selling Process**

**A close up of text on a white background

Description automatically generated**

Manage selling process mainly concern the record of a sale and the update of the quantities of different products in the store. The main subprocesses that are under the scope of this process are: display available products in the store, display products based on their type (Delicatessen / Sauces / Meat), add a product to the shopping list, delete a product from the shopping list, display the total amount due. It is linked to the database of products since it needs to update the quantities of the products sold to keep track of the existing inventory in the store. The following are the subprocesses of this process:

**\*Display Available Products in the Store:** This subprocess visualizes the quantities of the merchandise remaining in the stock.

**\*Add a Product to the Shopping List:** When the user will be recording a transaction, a shopping list will be created. Products that are sold can be added to this list. This is the role of this subprocess.

**\* Delete a Product to the Shopping List:** This subprocess allows the deletion of a product that was added to the shopping list.

**\* Display the Total Price:** Once the user finishes adding the products to the shopping list, this subprocess will allow him to display the total price of this list.

**5-3-5 Manage Warehouse and Factory Communication**

**A close up of a device

Description automatically generated**

Many information need to be communicated to the factory. This communication is done through emails sent automatically from the email of the shop to the email of the factory. This is the role of Manage Warehouse and Factory Communication. The subprocesses that are under the scope of this process are: sending need request for a product, receive a confirmation of product delivery, send request to return an expired product, and send inventory information of the products available in the warehouse. This process will only need access to the database of products from which it can get all the information needed to complete the subprocesses. The following are the subprocesses of this process:

**\*Send Need Request for a Product:** This subprocess checks the quantity of all products existing in the store. The user set a threshold for each product. If the quantity of the corresponding product went below this threshold, this process sends an email to the factory to inform them about it.

**\*Receive a Confirmation of Product Delivery:** This subprocess will show to the user the confirmation that the factory will send to confirm the delivery of an order. The confirmation will be shown in the system.

**\*Send Request to Return an Expired Product:** This subprocess have access to the database of products in the stock. If a product reaches its expiration date, it automatically send an email to the factory to inform them about it to get it back.

**\*Send Inventory Information to the Factory:** This subprocess summarizes the actual state of the warehouse ( the quantities of all products) in a PDF file and sends it to the factory.

**5-3-5 Manage Official Documents:**

**A screenshot of a cell phone

Description automatically generated**

Manage Official Documents is an important process in the functioning of the system. The main subprocesses that are covered by it are: generating a PDF formatted invoice for a chosen customer, generating a PDF formatted purchase order, generating a PDF formatted delivery form**,** storing invoice provided by the provider. This process has access to the database of transactions. The following are the subprocesses of this process:

\***Generate a PDF formatted invoice for a Chosen Customer:** This subprocess takes care of writing and preparing the invoice for a sale in a PDF format. The user selects the customer to who he wants to prepare the invoice. Then this subprocess creates a PDF format of the invoice.

\***Generate a PDF formatted Purchase Order:** This subprocess creates a PDF format of an order that the shop may make. The user will enter the corresponding transaction, then this subprocess create the appropriate purchase order in PDF format.

\***Generate a PDF formatted Delivery Form:** This subprocess creates a PDF format of a delivery form. Once a sale transaction is recorded, the user can use this subprocess to create the delivery from in an automated way.

\***Store the Invoice Provided by The Provider:** This subprocess allows the user to save the invoice that the supplier provides him with after a purchase transaction.

1. **Conceptual Architecture**

**6-1 Conceptual Architecture Diagram**

**6-2 Conceptual Architecture Description**

The Conceptual architecture diagram of VIRTUALSTORE presents all the process that this information system includes. Those processes are managing providers, clients, transactions, selling process, warehouse and factory communication, and official documents.

* **Managing Providers:** Consist of keeping track of the list of providers and their information, with the possibility of communicating to them information through an email.This is done through the adding, deleting, modifying, and sending email processes.
* **Managing Clients:** Consist of keeping track of the list of client and their information. This is done through the adding, deleting, modifying, processes.
* **Managing Transactions:** Consist of keeping track of the transactions realised by the store, either with client or with providers. This is done through the adding, deleting, modifying, processes.
* **Managing Selling Process:** Consist of providing the user with an interface to deal with the daily transactions. This is done through view products, add from the shopping list, delete from the shopping list, and display the total price processes.
* **Managing Warehouse and Factory Communication:** Consist of emitting and receiving information to and from the warehouse and factory. This is done through sending request, sending data, and receiving confirmation processes.
* **Managing Official Documents:** Consist of generating the essential official documents related to the processes mentioned in the context diagram, and storing them.

1. **Conceptual Data Models and Entity Relations**

**7-1 Description of the meeting with the client**

The purpose of our last meeting with Mrs. Mounacir was to elicit the requirements that we gathered. After, clarifying these requirements and making them actions ready for automation, we built the context diagram, conceptual process models, and the conceptual architecture of the system in order to represent clearly the functioning of the software. As we have seen in class with Dr. Kettani, the next step is to perform a data modeling of the system by building a data dictionary and determining entities. To achieve this, we need to gather all the forms that our client is using in his business. This operation is very helpful for performing data modeling. We contacted our client to send us the forms used in the shop and the stock. She sent us some of these forms that she has on her laptop, the remaining ones are in the shop. Due to the covid-19 virus outbreak in Casablanca, she couldn’t take the risk and go to the shop. Therefore, we worked only on the forms that we received. We had a call with Mrs. Mounacir to discuss some anomalies and problems that we faced in our first analysis of the forms. Mrs. Mounacir clarified everything to us and made us prepared to go further with the data modeling step.

**7-2 Presentation of the actual forms**

The forms that we gathered from the client are:

* **Liste des ventes:** this document is in a form of an excel sheet. It contains a column in which all products are written, a column in which the quantity sold is specified, and a column in which the price is computed (price of product\*quantity). At the bottom, the total sales revenue is computed.
* **Liste des achats:** this form is an excel document. It is a table of 4 columns. the first one contains the product purchased, the second one contains the name of the provider, the third one specifies the quantity purchased, and the fourth one contains the computed price (price of product\*quantity). At the bottom, the total of money paid in purchases is computed.
* **Suivi du stock:** this document is used to keep track of the quantities in the stock. This table helps to know the remaining products in the store and their value. At the bottom, the total value of the products in the stock is computed.
* **Caisse:** In this document, our client shows the flow of money in her business. The sales and purchases are visualized, and the net is computed.
* **Bon de livraison:** This documents serve as a proof for the company that client received his order, specifically the products mentioned in the designation and their respective quantity. After the client checks that the mentioned products are the one delivered and in the correct quantity, the client provides his signature in this form that the company will keep in case the client claims not to have received his order.
* **Bon de commande:**  This form serve in transmitting the information of the company’s order to their providers in order to get the provider confirmation of the order. Once the provider sign the form, he is obliged by the law to provide the order in the specified date.
* **Facture:** This form serve as a receipt of each transaction between ste Five Distrubution and their clients.

**7-3 Data Dictionary**

|  |  |  |  |
| --- | --- | --- | --- |
| Form | Name of Attribute | Description | C/NC |
| **Suivi de stock** | Designation | The product identifier | NC |
| Stock initial | Inventory before updating | NC |
| Stock restant | Inventory after updating | C |
| Sortie | Inventory changes (out) | NC |
| Prix HT | Price tax not included | NC |
| TVA | Tax percentage | NC |
| Total HT | Total amount tax not included | C |
| Montant TVA | Tax amount | C |
| Montant TTC | Total amount tax included | C |
|  |  |  |  |
| **Liste des achats** | Designation | The product identifier | NC |
| Fournisseur | The provider identifier | NC |
| Bon N° | Order form identifier | NC |
| Quantite | The product quantity | NC |
| Prix | Price per unit | NC |
| Total | Total price | C |
| Paye | Payed or not | NC |
| Date | The present date | NC |
|  |  |  |  |
| **Liste des ventes** | Designation | The product identifier | NC |
| Client | Client identifier | NC |
| Quantite | Porduct quantity | NC |
| Prix | Price per unit | NC |
| Total | Total price | C |
|  |  |  |  |
| **Caisse** | Date | The present date | NC |
| Recette | The total amount gathered | C |
| Especes | The total amount gathered in cash | C |
| Carte bq | The total amount gathered by credit card payments | C |
| Dépenses | Cash outflow designation | NC |
| Montant | Cash outflow amount | C |
| Reste | The difference between the total amount gathered (Recette) and the outflow amount (montant) | C |
|  |  |  |  |
| **Bon de livraison** | N° bon de livraison | The delivery form ID / identifier | NC |
| Adresse livraison | The address of delivery | NC |
| Personne à contacter | The contact information of the person to contact before the delivery | NC |
| N° BC | The order form identifier of the respective delivery | NC |
| Destinataire | The client | NC |
| Date de livraison | Delivery date | NC |
| Désignation | The products list | NC |
| Unité | Unit of measure of each product (KG / Unit / Liter …) | NC |
| Quantité | The corresponding quantity of the products in the specified unit of measure | NC |
|  |  |  |  |
| **Bon de commande** | Commande N° | The order form ID / identifier | NC |
| Expéditeur | The client | NC |
| Fournisseur | The provider | NC |
| Date de commande | Order date | NC |
| Adresse de livraison | Delivery address | NC |
| Adresse de facturation | Billing address (Where the bill should be sent if different than delivery address) | NC |
| Code d’article | Product ID / identifier | NC |
| Quantité | The products quantity | NC |
| Unité | The unit of measure of the quantity of the product | NC |
| Prix | The unit price | NC |
| Prix total | The total price | C |
| Date de livraison | Delivery date | NC |
|  |  |  |  |
| **Facture** | N° facture | Bill ID / identifier | NC |
| Paiement | Payment method | NC |
| Désignation | The products list | NC |
| Quantité | The product quantity | NC |
| Unité | The unit of measure of the quantity of the product | NC |
| Prix | The unit price of the product | NC |
| Prix total | The total price of order | C |
| Date | The date of the bill | NC |
| Client | The Client information | NC |

**7-4 Analysis of Data Dictionary: Identifying Anomalies**

* **Redundancy Identification:**

1. Designation: After going through the forms, it was found that this attribute is subject to redundancy. It exists in “Suivi de stock”, “Liste des achats”, “Liste des ventes”, “Bon de livraison”, “Bon de commande”.
2. Prix: after reading the forms, it was found that this attribute is redundant in all forms. It always means the price of the product concerned.

* **Synonyms Identification:**

1. The attribute “Total” in the form “Liste des ventes” and the attribute “montant” in the form “Caisse” are synonyms. Both of them refer to the amount of money paid by the customer after a sale.

* **Homonyms Identification:**

1. The attribute “Bon N°” exists in the form “Bon de commande” and “Bon de livraison”. However, it doesn’t refer to the same thing. In the first form, it refers to the corresponding number of the order. In the second form, it refers to the corresponding number of the delivery.
2. The attribute “Date” exists in “Liste des achats”, “Liste des ventes”, “Bon de commande”, and “Bon de livraison”. In the first one, it refers to the date of purchase, while in the second one it refers to the date of sale. In the form “Bon de commande”, it means the date of order, while in “Bon de livraison” it means the date of delivery.
3. The attribute “Total” exists in the form “Liste des achats” and “Liste des ventes”. However, they do not refer to the same thing. In the first form, it refers to the total amount to be paid in a purchase. In the second form, it refers to the total amount received after a sale.
4. The attribute “Quantité” exists in “Liste des achats” and “Liste des ventes” even if it doesn’t refer to the same thing. In the first form, it refers to the quantity of product purchased, while in the second one it means the quantity of product sold.

**7-5 Demonstration of the Completeness of Data**

As our professor Dr. Kettani always says, moving to the next step without checking the completeness of the previous one will have a negative effect on all the remaining steps. In order to avoid any mistake or error while developing the system, it is primordial to check the completeness of the data we gathered, before we start determining the entities that can fit into the system. To achieve this, we reviewed all data existing in the forms we gathered, and we made sure that they are clear from our side and from the client’s side. Furthermore, we had an additional call with our client Mrs. Mounacir in which we went through all the descriptions stated in the data dictionary. Mrs. Mounacir confirmed that the descriptions are right and she could not find a data processed in her business that we did not included in the data dictionary. Also, she confirmed the different anomalies that we detected when analyzing the data dictionary. Therefore, we can affirm that the data is complete, and we can move to determining the entities and building the ER-diagram.

**7-6 Description and Justification of the Use of Entities**

* **Client: :** Constitutes an entity in our relational model. Client is considered as an entity because it can stand alone within the context of the system. It is formed by several attributes that uniquely characterizes a client. These attributes are: Client Id, Company name, Liaison name, Liaison phone number, Company phone number, address, email address.
* **Provider:** is an entity in our entity-relation model. Provider is an entity since it is an element of the system context that is independent and has several attributes that distinguish it. These attributes are: Provider Id, Company name, Liaison name, Liaison phone number, Company phone number, address, email address, Type of product delivered.
* **Sale:** is an entity within our ER-diagram. It is an independent element of the system that stands alone. It has many attributes that characterize it particularly. These attributes are: Sale Id, Amount of money received, Amount of money receivable, Total amount, Payment mode.
* **Purchase:** constitutes an entity in our entity relational model. Purchase can stand alone within the context of our system. Many attributes constitute this entity: Purchase Id, Amount of money paid, Amount of money payable, Total amount, Payment mode.
* **Product:** is an entity in our entity-relation model. Product is an entity because it stands alone within the context. The attributes that uniquely define it are: Product Id, Product name, Product category, Quantity in the stock, Price tax not included, TVA percentage, Price taxes included, Expiration date, Threshold quantity.
* **Bill:** is an entity within our ER-diagram. The attributes that particularly characterize this entity are: Bill Id, Payment mode, total amount, date.
* **Delivery:** constitutes an entity in our entity-relation model. This entity is a stand-alone element in the system context. The attributes that define it are: Delivery Id, Date of delivery, address of delivery, address of billing, Total amount.
* **List-in:** is an entity within our ER-diagram. The attributes that constitute uniquely this entity are: Quantity, Sub-total price.
* **List-out:** is an entity in our entity-relation diagram. The attributes that particularly define it are: Quantity, Sub-total price.

We found in some of our relationships, between some entities, that they are M:N. We learned in class that this kind of relationships is difficult to deal with, which means that we need to get rid of them. One of the solutions is to build a bridge entity to break these relationships into a 1:N relationship. Here are the relationships that needed a bridge:

* **The relation between Purchase and Product:** a product can be involved in many purchases and a purchase can involve many products, which makes the cardinalities are M:N. To get rid of this M:N relation, we use the **List-in** entity. This entity contains the attributes Quantity (quantity of the product), and the sub-total price (Price of the product\*quantity). A List-in can be associated to many products, but a product can only be associated to one List-in. For purchase, it can be related to many List-in, but a List-in is specific to one purchase.
* **The relation between Sale and Product:** a product can be involved in many sales and a sale can involve many products, which makes the cardinalities are M:N. To get rid of this M:N relation, we use the **List-in** entity. This entity contains the attributes Quantity (quantity of the product), and the sub-total price (Price of the product\*quantity). A List-in can be associated to many products, but a product can only be associated to one List-in. For sale, it can be related to many List-in, but a List-in is specific to one sale.

**7-7 Entity Relations Diagram**

A close up of a map

Description automatically generated

**7-7-1 Description of the Entity Relation Diagram**

An entity relation diagram is a graphical representation of all entities that will exist in the system along with the relationships between them. The entities that exists in the diagram are the ones described above.

We have designed the interaction between Sale and product to be done through the bridge entity List-out. This latter has as attributes: Quantity and Sub-total price. This bridge entity allows to define a quantity and a total price by product in a way that will keep the entity-relation diagram consistent. Each product can be associated to many List-out, but this latter can only be related to one product. Also, sale can be associated to many List-out in case many products were sold, but each List-out is related only to one sale. This justifies the cardinalities written next to these entities in the diagram

We have designed the interaction between purchase product to be done through the bridge entity List-in. This latter has as attributes: Quantity and Sub-total price. This bridge entity allows to define a quantity and a total price by product in a way that will keep the entity-relation diagram coherent and cohesive. Each product can be associated to many List-in, but this latter can only be related to one product. Also, purchase can be associated to many List-in in case many products were sold, but each List-in is related only to one sale. This justifies the cardinalities written next to these entities in the diagram.

The cardinalities related to the interaction between sale and client are reasonable. Each sale is related to one and only one client. However, a client can do many sales, which makes him related to lot of sales. The same thing applies for the interaction between provider and purchase. Each purchase can be done from one provider, while the shop can purchase many times from the same provider, which justifies the cardinalities associated to this interaction.

The bill entity is associated to Sale, Purchase, and Delivery. The bill can contain many sales and purchases, but they are billed only once, which justifies the cardinalities associated to them. The bill contains many transactions that can correspond to different deliveries. Since the bill is the entity associated to these transactions, it can have many deliveries. However, a delivery is only recorded in one bill.

**7-8 Confrontation of Data Models with Process Models**

For the last step of the CDM process, we will be confronting the Data Model and the Process Model. We will show that each entity in the CDM is used, at least, by one of the CPMs.

|  |  |
| --- | --- |
| **Data** | **The Process** |
| Product\_ID | Manage Selling process / Manage Warehouse and Factory Communication |
| Product\_Name | Manage Selling process / Manage Warehouse and Factory Communication |
| Product\_Category | Manage Selling process / Manage Warehouse and Factory Communication |
| Quantity\_in\_Stock | Manage Selling process / Manage Warehouse and Factory Communication |
| Price\_no\_Tax\_included | Manage Selling process / Manage Warehouse and Factory Communication |
| Tax\_Percentage | Manage Selling process / Manage Warehouse and Factory Communication |
| Price\_Tax\_Included | Manage Selling process / Manage Warehouse and Factory Communication |
| Expiration\_Date | Manage Selling process / Manage Warehouse and Factory Communication |
| Threshold\_Quantity | Manage Selling process / Manage Warehouse and Factory Communication |
| Sale\_ID | Manage Selling Process / Manage Transactions |
| Amount\_Received | Manage Selling Process / Manage Transactions |
| Amount\_Receivable | Manage Selling Process / Manage Transactions |
| Total\_Amount | Manage Selling Process / Manage Transactions |
| Payment\_Mode | Manage Selling Process / Manage Transactions |
| Purchase\_ID | Manage Transactions |
| Amount\_Paid | Manage Transactions |
| Amount\_Payable | Manage Transactions |
| Client\_ID | Manage Client |
| Company\_Name | Manage Client / Manage Provider |
| Liaison\_Name | Manage Client / Manage Provider |
| Liaison\_Phone | Manage Client / Manage Provider |
| Company\_Phone | Manage Client / Manage Provider |
| Address | Manage Client / Manage Provider |
| Email\_Address | Manage Client / Manage Provider |
| Provider\_ID | Manage Provider |
| Type\_Provided\_Product | Manage Provider |
| Delivery\_ID | Manage Official Documents |
| Date\_Delivery | Manage Official Documents |
| Address\_Delivery | Manage Official Documents |
| Address\_Billing | Manage Official Documents |
| Total\_Amount\_Shipping | Manage Official Documents |
| Bill\_ID | Manage Official Documents |
| Final\_Total\_Amount | Manage Official Documents |
| Date\_Billing | Manage Official Documents |
| Quantity | Manage Official Documents / Manage Transactions / Manage Selling Process |
| Sub\_Total\_Price | Manage Official Documents / Manage Transactions / Manage Selling Process |

1. **Conclusion**

After going through the Requirement Engineering Document (RED), we now know how to convert a real-life problem that is complex and abstract to a simplified model. We tried to adopt the best process that suits since we know that a good product or end results depends on the quality of the process. We tried to break our problem into small sub problems each having a normal difficulty allowing us to solve each one of them easily. In short, we used the steps of software engineering to build a system that highlights the network of paths that rely the building blocks of the project. Nevertheless, if we say that we are done here then we are wrong, because we still need to go through another important step that is as crucial to the development of a good product which is the design step where we dig deeper and exploit the flaws of our system to make it better by having a good data model and staying as far away from anomalies.

To make thing clearer, we will bring into the scene the technology stack or enablers that we are going to rely on to build our software