People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of the requirements for the Degree of

MASTER

In **Electronics**

Option: **Computer Engineering.**

Title:

Building a Management System for IGEE's General Means Store.

Presented by:

– Djamel Eddine Bendjedia.

Supervisor:

Dr. A. Zitouni

Registration Number: ..../2021

ABSTRACT

The project proposes technical solutions to problems related to IGEE's general means store management. The aim of this project is to optimize the management of the store and to facilitate the work through creating an interactive web application that runs on a web server in order to simplify the work and automate the management of the store.

Analysis of the current management process and tasks of the store employee is necessary for enhancing the management process of the store.

The web application helps in avoiding mistakes and reducing time and efforts. It will automate tasks of the storekeeper related to products, their categories and their transaction records.

This project has been carried out using the design and development processes. The design had been handled using the Unified Modeling Language (UML). For the implementation of this web application, several web development tools and skills had been used: HTML, CSS, PHP, and MariaDB.

Keywords: CSS, **database**, **HTML**, **inventory**, **management system**, **MariaDB**, **PHP**, **store**, **web application**.

DEDICATION

I dedicate this modest work to my dear parents, my brothers, and to all my family and friends.

Djamel Eddine.

ACKNOWLEDGMENTS

I would like to thank my project supervisor and my teacher Dr. A. Zitouni for his assistance, advices, and valuable feedback for the realization of this senior project.

I wish to thank Dr. Khalifa for his valuable course in Database Management System which was useful and helpful.

I want also to thank my family members and my friends whom help, encourage, and support me during my work.

LIST OF FIGURES

Figure 1: The Components of a URL. 5
Figure 2: Request Response Cycle and Domain to IP Address Translation [15]. 5
Figure 3 MMD Structure Diagram. 8
Figure 4: Admin UCD. 17
Figure 5: User UCD. 17
Figure 6: Authentication Use Case Description. 18
Figure 7: Logout Use Case Description. 18
Figure 8: DisplayCategories Use Case Description. 19
Figure 9: ManageCategories Use Case Description. 19
Figure 10: DisplayStockType Use Case Description. 20
Figure 11: ManageStockType Use Case Description. 20
Figure 12: DisplayProducts Use Case Description. 20
Figure 13: ManageProducts Use Case Description. 21
Figure 14: DisplayStockFlow Use Case Description. 21
Figure 15: ManageFlow Use Case Description. 22
Figure 16: ProductStockLog Use Case Description. 22
Figure 17: ManageAdminsUsers Use Case Description. 23
Figure 18: AddCategory Use Case Description. 23
Figure 19: UpdateCategory Use Case Description. 24
Figure 20: AddStockType Use Case Description. 24
Figure 21: UpdateStockType Use Case Description. 25
Figure 22: AddProduct Use Case Description. 25
Figure 23: UpdateProduct Use Case Description. 26
Figure 24: AddFlow Use Case Description. 26
Figure 25: Edit Use Case Description. 27
Figure 26: AddUser Use Case Description. 27
Figure 27: UpdateAccount Use Case Description. 28
Figure 28 DisplayProducts Use Case Description. 28
Figure 29: UpdateAccount Use Case Description. 28
Figure 30: Filter Use Case Description. 29
Figure 31: Sort Use Case Description. 29
Figure 32 Schema Diagram of the Database igee_gms 31
Figure 33 Database Relations and Relationships 32
Figure 34 Architecture of the Application 35
Figure 35: Login Interface 36
Figure 36: Home Page Interface 37
Figure 37: Categories Interface 37
Figure 38: Manage Categories Interface 38
Figure 39: Stock Type Interface 38
Figure 40: Manage Stock Type Interface 39
Figure 41: Products Interface 40
Figure 42: Manage Products Interface 41
Figure 43: Stock Flow Interface 42

Figure 44: Manage Flow Interface 43
Figure 45: Product Log Interface.. 44
Figure 46 Admin Interface 45
Figure 47: Products Interface 46
Figure 48: Mobile Login Interface. 47
Figure 49: Manage Products Mobile Interface 48

LIST OF TABLES

Table 1: Arrows Used in Use Case Diagram’s Relationships 16
Table 2: Data Description 34

LIST OF ABBREVIATIONS

GMS	General Means Store
IIS	Internet Information Services
HTTP	HyperText Transfer Protocol
HTML	HyperText Markup Language
URL	Uniform Resource Locator
HTTPS	HyperText Transfer Protocol Secure
IP	Internet Protocol
DNS	Domain Name System
CSS	Cascading Style Sheets
SSL	Secure Sockets Layer
TLS	Transport Layer Security
PHP	PHP: Hypertext Preprocessor
IGEE	Institut de Génie Electrique et Electronique
SQL	Structured Query Language
UML	Unified Modeling Language
MMD	Means and Maintenance Department
MD	Means Department
CRUD	Create Read Update Delete
UCD	Use Case Diagram
RDBMS	Relational Database Management System
CMD	Command Prompt
CGI	Common Gateway Interface
IBM	International Business Machines
ISO	International Organization for Standardization
ANSI	American National Standards Institute
DML	Data Manipulation Language
DDL	Data Definition Language
DCL	Data Control Language
OMG	Object Modeling Group
ID	Identification/Identity/Identifier
GMSMS	General Means Store Management System

TABLE OF CONTENTS

ABSTRACT I

DEDICATION II

ACKNOWLEDGMENTS III

LIST OF FIGURES IV

LIST OF TABLES VI

LIST OF ABBREVIATIONS VII

TABLE OF CONTENTS VIII

GENERAL INTRODUCTION X

CHAPTER ONE: AN OVERVIEW OF THE PROJECT AND WEB APPLICATIONS 1

1.1 INTRODUCTION: 1
1.2 SUBJECT PRESENTATION: 1
1.2.1 Problem Statement: 1
1.2.2 Project Objectives: 2
1.3 WEB APPLICATIONS: 2
1.3.1 Definition: 2
1.3.2 Web Applications Advantages: 2
1.3.3 Static Website Versus Dynamic Website: 3
1.3.4 The Internet and The Web: 3
1.3.5 Web Access and Web Servers: 3
1.3.6 Overview of HTTP and HTTPS: 4
1.3.7 Webpage Address (URL): 4
1.3.8 Frontend and Backend Development 6
1.3.9 Responsive Web Design: 7

CHAPTER TWO: OVERVIEW OF THE MEANS AND MAINTENANCE DEPARTMENT 8

2.1 INTRODUCTION: 8
2.2 ABOUT MEANS AND MAINTENANCE DEPARTMENT (MMD): 8
2.3 THE DEPARTMENT OF MEANS (MD): 8
2.4 GENERAL MEANS STORE: 9
2.5 MEMBERS OF THE MD: 9
2.6 STAFF TASKS IDENTIFICATION: 9
2.6.1 Head of MMD and Responsible of MD: 9
2.6.2 The Storekeeper: 9

CHAPTER THREE: DEVELOPMENT TOOLS AND TECHNOLOGIES 11

3.1 INTRODUCTION: 11
3.2 DEVELOPMENT PLATFORM AND TOOLS: 11
3.2.1 XAMPP: 11
3.2.2 Apache HTTP Server: 11
3.2.3 MariaDB: 11
3.2.4 phpMyAdmin: 12
3.2.5 Windows Command Prompt: 12

3.2.6 Visual Studio Code: 12
3.2.7 Browser: 13
3.2.8 StarUML: 13
3.3 PROGRAMMING LANGUAGES: 13
3.3.1 PHP: 13
3.3.2 SQL: 14
3.3.3 HTML: 15
3.3.4 CSS: 16

CHAPTER FOUR: DESIGN 17

4.1 INTRODUCTION: 17
4.2 DESIGN REQUIREMENTS: 17
4.2.1 Modeling Language: 17
4.2.2 Unified Modeling Language: 18
4.2.2.1 Definition: 18
4.2.2.2 Types of UML Diagrams: 18
4.3 MODELING: 18
4.3.1 System Actors: 18
4.3.1.1 Admin: 19
4.3.1.2 User: 19
4.3.2 Use Cases: 19
4.3.3 Use Case Diagrams: 15
4.3.3.1 Admin Use Case Diagram: 17
4.3.3.2 User Use Case Diagram: 17
4.3.4 Textual Description of Use Cases: 18

CHAPTER FIVE: IMPLEMENTATION 30

5.1 INTRODUCTION: 30
5.2 DATABASE: 30
5.2.1 Database Schema: 30
5.2.2 Database Relations and Their relationships: 32
5.2.3 Data Description: 33
5.3 THE ARCHITECTURE OF THE APPLICATION: 34
5.4 INTERFACING THE APPLICATION: 35
5.4.1 Login Interface: 36
5.4.2 The Admin Area: 36
5.4.3 The Semipublic Area: 45
5.4.4 Mobile Interface: 46

GENERAL CONCLUSION 49

FUTURE ENHANCEMENTS 50

REFERENCES 51

GENERAL INTRODUCTION

The development of technologies and science is increasing day after day, and this is reflected on human life by simplifying their life in different domains. Why not? And the world today became a small village as a result of newly invented technologies. One of the important inventions that plays a big role in facilitating the life of humans is the computer and its related technologies.

Information technology (IT) is the use of technologies to handle business issues that companies face in their business. One of the main component of IT is the computer which is nowadays used as a powerful tool to solve business issues and problems. Computer is the safest way to store, analyze, process and backup data in order to simplify data and business management.

Inventories are one of the area which encounter problems in management due to the manual methods used. Hence, developing a system to handle management tasks and solve these problems is necessary. Each Faculty has its general product store which is used to store the purchased products and supply the faculty with necessary means. The project aims to design and develop an easy to use web application that helps in the management process of the general means store of our institute.

The report explains the process in which the web application helps in simplifying the management process and tasks of the store employee. Moreover, it describes the tools and techniques used in designing, developing and implementing this application.

Chapter one introduces an overview of the project and web applications. In **The second chapter,** The Department to which the general means store is related is described briefly including its staff and store. Furthermore, **chapter three** covers the development and technologies used in building the web application. It presents the programming languages and the development platform that had been used in creating the application. The design stage of the web application is covered in the **fourth chapter** which shows that it has been handled using the unified modeling language (UML) and its useful tools. **Chapter five** includes the implementation of the developed web application. Additionally, it covers an interfacing to the developed application. The report ends with a general conclusion.

CHAPTER ONE: AN OVERVIEW OF THE PROJECT AND WEB APPLICATIONS

1.1 Introduction:

In this chapter, I introduce an overview on the project by presenting its objectives and the problems that it aims to solve. An introduction about web applications is covered by presenting some related generalities which help in understanding web applications.

1.2 Subject Presentation:

The subject of the project is to design and create a web application that help the management of the general means store (GMS) of IGEE. The application should solve some problems encountered by the department staff in managing this store.

1.2.1 Problem Statement:

After discussing with some members of the department of GMS including the head, I got a clear vision about the work of the department and the tasks of the store employee which allowed me to get a clear idea about how the application will be implemented and about the problems related to the store management. Deduced problems from the interview are stated as follows:

- Manual management of the stock.
- Lack of application or software helps in managing the store.
- The department head has no tool helps him to see and check instantly the store status from its office. He need to go to the store to know or call the store employee
- Tasks related to the storekeeper are done manually using documents.
- Traditional and manual way in archiving previous data.
- Probability of losing data of the archive since it is a traditional one.
- Space consumed by archive.

1.2.2 Project Objectives:

After identifying problems related to the GMS management, the project should provide solutions to these problems through achieving the following objectives:

- Development of an easy to use web application that handles the management tasks and facilitates the work of the storekeeper.
- Automation and digitization of the data processing.
- Digitization of the store archive through building a database for the store.
- Provide an interface for monitoring the store.
- Record incoming (bought) and outgoing delivered products with all necessary data of the transaction.
- Making data access easy.

1.3 Web applications:

1.3.1 Definition:

A web application is a software that is accessed using a web browser by communicating with the web server [1] on which the application resides somewhere. It could only be accessed through an Internet connection, thus, the term "web" became quite applicable as well [2]. Web applications can be created using frontend and backend languages [3].

1.3.2 Web Applications Advantages:

- Quick and Easy Installation and Maintenance.
- Easily Customizable.
- Ability to Incorporate Modern Design Schemes.
- Accessible Anywhere [4].
- Portable and Platform Independent.

1.3.3 Static Website Versus Dynamic Website:

Static website is the basic type of websites that is easy to create. Its web pages are coded in HTML. The codes are fixed for each page, so the information contained in the page does not change and it looks like a printed page [5]. This type of website can be considered as a read only website, thus there is no significant interaction with the user.

However, dynamic website is a collection of dynamic web pages whose content changes dynamically. It accesses content from a database. Therefore, when the content of the database is altered or updated, the content of the website is also altered or updated. Dynamic website uses client side scripting or server-side scripting, or both to generate dynamic content [5]. Web applications are a good example for dynamic websites.

1.3.4 The Internet and The Web:

The internet is an international network of connected computers in purpose of sharing information. It is a cooperative effort governed by a system of standards and rules.

On the other hand, the web is just one of the ways information can be shared over the internet. It uses a protocol called HTTP (HyperText Transfer Protocol). The web is a subset of the internet. It is just one of many ways information can be transferred over networked computers [6].

1.3.5 Web Access and Web Servers:

People access websites using software called a web browser. Popular examples include Firefox, Internet Explorer, Safari, Chrome, and Opera. In order to view a web page, users might type a web address into their browser, follow a link from another site, or use a bookmark [7]. Web browser is also called web client [6].

When the user ask the browser for a webpage, the request is sent across the Internet to a special computer known as a web server which hosts the website. Web servers are special computers that are constantly connected to the Internet, and are optimized to send web pages out to people who request them [7]. More accurately, the web server is the software -not the computer- that allows the computer to communicate with other computers on the web by allowing them to handle HyperText Transfer Protocol (HTTP) transactions. However, it is common to use the word “server” to refer to the computer. Popular examples of web servers are Apache and Microsoft Internet Information Services (IIS). Web servers are also called HTTP servers [6].

When the server receives the client request, it starts processing the request by bringing the requested page. In case the requested page contains a server side script language, the server will interprets this script and returns the final output in HTML code only to the browser.

1.3.6 Overview of HTTP and HTTPS:

The requests and responses mentioned earlier are handled by HTTP protocol [6]. It is meant by a protocol the set of rules and standards governing the data transfer between the client and the server.

HTTP is the basis for the World Wide Web [8] and is implemented in two programs: a client program and a server program. The client program and server program talk to each other by exchanging HTTP messages. HTTP defines the structure of these messages and how the client and server exchange data and information [9].

HTTPS is the secure web protocol thus, S stands for secure. It encrypts traffic using SSL or TLS security protocols to protect the data when being transmitted [10].

1.3.7 Webpage Address (URL):

Every web resource has its own address called URL. URL stands for Uniform Resource Locator [6]. A URL is composed of different parts, some are mandatory and others are optional [11]. Most important parts of a URL are shown in Figure 1.

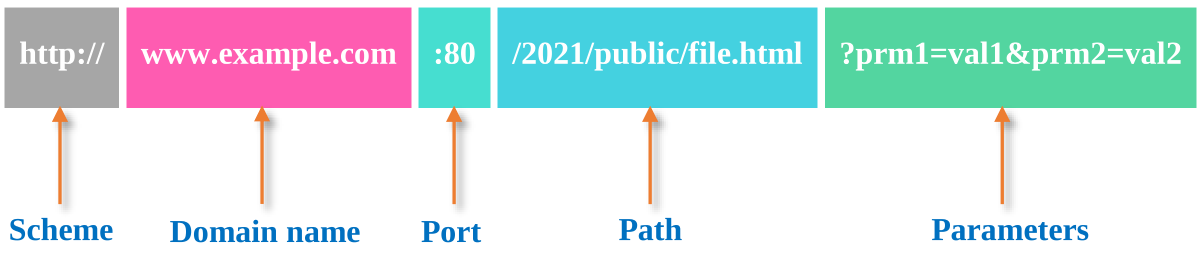


Figure 1: The Components of a URL.

The scheme indicates the protocol that the browser must use to request the resource. Usually, the protocol for websites is HTTPS or HTTP [11].

A domain name is a human friendly string of text that maps to a numeric IP address, used to access a website from a client software [12]. IP address is a unique number assigned to every device connected to a network that uses the Internet protocol [13] [14]. Furthermore, it is the DNS which provides for the browser the IP address associated to the requested website based on the domain name. See Figure 2. DNS acts like a phonebook that associates the domain with its IP address [7].

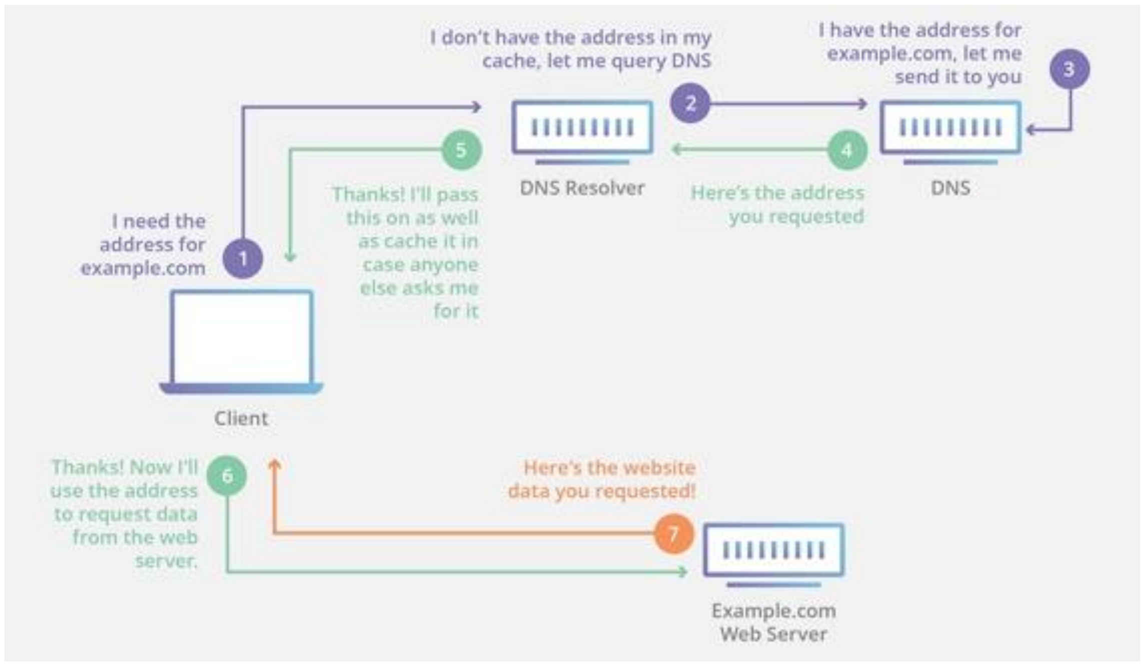

I don't have the address in my cache, let me query DNS
I have the address for example.com, let me send it to you
3
2
5
4
DNS Resolver
DNS
Thanks! I'll pass this on as well as cache it in case anyone else asks me for it
Here's the address you requested
I need the address for example.com
1
Client
Thanks! Now I'll use the address to request data from the web server.
6
Here's the website data you requested!
7
Example.com Web Server

Figure 2: Request Response Cycle and Domain to IP Address Translation [15].

The port indicates the technical gate used to access the resources on the web server. It is usually omitted if the web server uses the standard ports of the HTTP protocols (80 for HTTP and 443 for HTTPS). Otherwise, it is mandatory [11].

/2021/public/file.html is the absolute path through directories on the server to the requested resource, in our example it is the HTML document, **file.html**. [6].

Parameters used in the URL are extra information sent to the web server which will use them in processing a server side language script before sending the requested file. This portion is a list of pairs of a parameter and its assigned value [11]. For example: the server will use them in retrieving data from the database.

Thus, the URL in Figure 1 uses the HTTP protocol to connect to a web server on the internet called **www.example.com** to request the document **file.html** located in public directory which is in 2021 directory [6].

1.3.8 Frontend and Backend Development

Web development falls under two broad categories: frontend development and backend development.

Frontend development refers to any aspect of the design process that appears in or relates directly to the browser. That includes HTML, CSS, and JavaScript.

On the other hand, backend development focus on the server, including applications and databases that run on it. It requires the knowledge of server side programing language such as PHP, python, Ruby and other server side programing languages in order to create applications that provide the functionalities required by the site. These applications handle tasks and features like forms processing, content management systems, and online shopping. Additionally, familiarity with database configuration and languages is needed to store the site data. Some common database languages include MySQL, Oracle, and SQL Server [6].

Backend development is the development process that is related to the server. It is hidden from the user and executed in the server, thus the name backend.

1.3.9 Responsive Web Design:

Responsive Web Design is a strategy for providing appropriate layouts to devices based on the size of the viewport (browser window). The key to Responsive Web Design is serving a single HTML document (with one URL) to all devices, but applying different style sheets based on the screen size in order to provide the most optimized layout for that device. For example, when the page is viewed on a smartphone, it appears in one column with large links for easy tapping. But when that same page is viewed on a large desktop browser, the content rearranges into multiple columns with traditional navigation elements [6].

CHAPTER TWO: OVERVIEW OF THE MEANS AND MAINTENANCE DEPARTMENT

2.1 Introduction:

This chapter presents the department that the project tries to implement solutions to. It deals with the parts of the department and their roles. In addition, it describes the members of the department and identify their tasks.

2.2 About Means and Maintenance Department (MMD):

The means and maintenance department is considered as one of the most important departments in the institute that aims to maintain a good process and cleanliness of the institute [16]. Its role is to provide the institute with necessary means and handle maintenance tasks.

The department is composed of two departments (see figure 3): department of means and department of maintenance. Each has a responsible and they are all governed by the head of the means and maintenance department. The project is related to the department of means since it builds a web application that help this department in the management process.

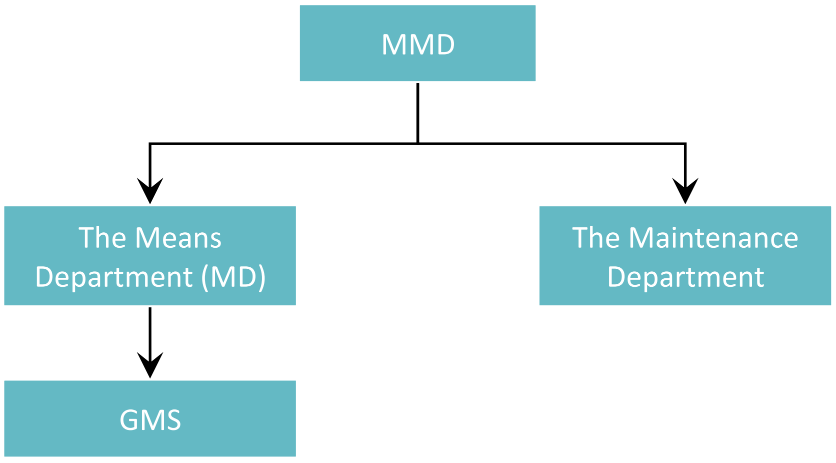


Figure 3 MMD Structure Diagram.

2.3 The Department of Means (MD):

The means department is the department that provides products to the institute departments. It has the following tasks:

- Ensure payments of financial credits specified to different purchases [16].
- Ensure Good Management and Monitoring of the GMS.
- Supply of different institute departments with the required means.
- Dealing with suppliers and make deals with them.
- Creating orders and reception receipts.

2.4 General Means Store:

GMS is the store belonging to the department of means. It is used to store the purchased means in order to supply the institute departments with the required means as well as keeping the means in a good state.

2.5 Members of the MD:

The members of the means department are listed below:

- The head of the means and maintenance department.
- The responsible of the means department.
- The storekeeper.

2.6 Staff Tasks Identification:

This section defines the tasks of the MD members. The head of the MMD and the responsible of the MD share the same tasks since the head has high privileges.

2.6.1 Head of MMD and Responsible of MD:

- They ensure payments of financial credits specified to different purchases.
- They manage and monitor the GMS.
- They deal with suppliers and make deals with them.
- They deal with reception and orders receipts.

2.6.2 The Storekeeper:

The storekeeper has the following tasks:

- He organizes the means in the store based on the corresponding category.

- He controls the products (means) and checks expired products to be omitted.
- He receives purchased products and verify them in term of quality and safety from defects.
- He maintains the required conservation and storage conditions.
- He informs the administration about products that are out of stock.
- He handles products orders and record all products exchanges and transactions.
- He Organizes archive of previous transactions or deliveries.

CHAPTER THREE: DEVELOPMENT TOOLS AND TECHNOLOGIES

3.1 Introduction:

It is mandatory for any developer wants to develop an application to use a development platform that provides the necessary tools which help him in developing the intended application. Development platforms vary depending on the programing languages and technologies to be used. This chapter introduces briefly the platform, tools and programing languages that had been used in the Development of this web application.

3.2 Development Platform and Tools:

3.2.1 XAMPP:

The XAMPP term can be unofficially broken down as: cross platform, **A**pache, **M**ariaDB, **P**HP and **P**erl. It is a free and open source cross platform web server solution stack package developed by Apache Friends which consists mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages [17].

3.2.2 Apache HTTP Server:

The Apache HTTP Server was launched in 1995 as an outcome of the Apache HTTP Server Project that aimed to develop and maintain a free and open-source HTTP server for modern operating systems. It has been the most popular web server on the Internet since April 1996 [18].

3.2.3 MariaDB:

MariaDB is a fork of the MySQL relational database management system (RDBMS) as an improved version. The MariaDB RDBMS comes with usability, security enhancement, performance improvements and numerous inbuilt powerful features that are not found in MySQL [19]. It is made by the original developers of MySQL -as enhanced and replacement for MySQL- and guaranteed to stay open source. MariaDB provides an SQL interface for accessing data [20].

3.2.4 phpMyAdmin:

phpMyAdmin is a free and portable web application tool written in PHP, intended to handle the administration of MySQL and MariaDB over the Web [21] [22]. It supports a wide range of operations on MySQL and MariaDB. Frequently used operations can be performed via the user interface, while still having the ability to directly execute any SQL statement [21].

3.2.5 Windows Command Prompt:

The Command Prompt, officially called the Windows Command Processor and often abbreviated to CMD, is the command line interface for Windows operating systems. A command line interface is a way of interacting with a computer directly using text commands [23]. The command prompt was used to perform SQL queries and CRUD operations on the database.

3.2.6 Visual Studio Code:

Visual Studio Code is a lightweight but powerful free source code editor which runs on desktops having Windows, macOS or Linux operating system installed on. It comes with built-in support for JavaScript, TypeScript and Node.js. Besides, it has a rich ecosystem of extensions for other languages like C++, C#, Java, Python, PHP and Go [24]. It also provides runtimes such as .NET and Unity. It allows the developer to code in any programming language without switching the editor since it has support for many languages [25].

Visual studio code provides for the developer helpful features such as keywords highlighting, IntelliSense and Auto code completion suggestions [25].

The IntelliSense features was so helpful during writing the code of the application. It is a general term for various code editing features including: code completion, parameter info, quick info, and member lists. IntelliSense features are sometimes called by other names such as "code completion", "content assist", and "code hinting" [26].

3.2.7 Browser:

A web browser is the application people use to view pages and navigate the World Wide Web [27]. The core purpose of a web browser is to connect to web servers, request documents, and then properly format and display those documents. Web browsers can also display files on your local computer, download files, and in some cases even allow the user to send and retrieve email. What the browser is best at, however, is retrieving and displaying web documents [28].

When the web browser fetches data -almost written in HTML and CSS- from an internet connected server, it uses a piece of software called a rendering engine (also called browser engine or layout engine) to translate that data into text and images. Web browsers read this code of HTML and CSS to create what we see, hear and experience on the internet [29] [6].

3.2.8 StarUML:

StarUML is a sophisticated software modeler aimed to support agile and concise modeling. It is an open source software that supports the UML (Unified Modeling Language) framework for system and software modeling [30] [31].

3.3 Programming Languages:

3.3.1 PHP:

Rasmus Lerdorf wrote the Personal Home Page (PHP) programming language as a way to improve how his Common Gateway Interface (CGI) scripts worked. After some encouragement and help, PHP morphed into its own programming language and a new name PHP: Hypertext Preprocessor [32].

PHP is a recursive acronym that stands for PHP: Hypertext Preprocessor. It is a free, popular, open source and powerful server side programing language [32] [33]. PHP is a server side programing language because it is processed and executed in the server of the website, thus it is hidden from the website users since its execution happens behind the scene.

PHP is popular for building dynamic web applications. It is a scripting language designed specifically for use on the web, with features that make web design and programming easier [33].

Using PHP, the developer has unlimited control over the web server. Whether he need to modify HTML on the fly, process a credit card, add user details to a database, or fetch information from a third-party website, he can do it all from within the same PHP files in which the HTML itself resides. In addition, it can communicate with the database to store and retrieve data easily [34].

To embed PHP within HTML we have to use .php files to write our webpages. Then, the server will take in consideration that it is a PHP file. Hence, it will execute php scripts before returning the resulted HTML code as a response to the browser request. During PHP execution the server may need to access the database couple of times if it is required by the script.

3.3.2 SQL:

SQL stand for Structured Query Language. It is the standard language used to communicate with a relational database by querying and manipulating data and defining structures in it [35] [36]. Initially developed at IBM in the early 1970s, SQL became an ANSI and ISO standard in 1986.

SQL is a powerful, and a simple language, but it can do many things, such as execute queries, retrieve, insert, update, and delete data, create databases and tables, and much more [35].

There are three major components of the SQL language. The first is called Data Manipulation Language (DML). This module of the language allows the developer to retrieve, update, add, or delete data in a database. The second component is called Data Definition

Language (DDL). DDL enables the developer to create and modify the database itself. For example, DDL provides ALTER statements, which allows to modify the design of tables in a database. Finally, the third component, Data Control Language (DCL), maintains proper security for the database [37].

3.3.3 HTML:

Web pages are documents, much like the document that are created in a word processor like Microsoft Word. To read a word processor document we use software like Microsoft Word, which knows how to open, read, and parse documents formatted or laid out in a certain way so that the various headings, spacing, and other elements of that document appear as intended. In addition, if the document is sent to another user having Microsoft Word, he can open and see the document in the same way. The reason behind this is that there are hidden formatting elements embedded in the document that tell Microsoft Word how to format or layout and display the text seen on the page. Similarly, HTML provides hidden formatting and layout information for webpages that tell the web browser how to display them [33].

HTML stands for HyperText Markup Language. The HyperText part refers to the fact that HTML allows developer to create links that allow visitors to move from one page to another quickly and easily. Markup language allows to annotate text and these annotations provide additional meaning to the content of a document. HTML code is added around the original text to be displayed and the browser then uses this code to display the page correctly. This added code is called tag [7].

HTML defines a set of common elements for web pages: headings, paragraphs, lists, and tables. It also defines character formats such as boldface and code examples. These elements and formats are indicated inside HTML documents using tags [28].

3.3.4 CSS:

Although there are default styles associated with HTML tags in the aim of describing the structure of a document, HTML doesn’t say much about how a page looks when it’s viewed. Cascading Style Sheets (CSS) enable the developer to apply advanced formatting to HTML tags. They are a way to control how the browser renders HTML elements. [28].

CSS is a simple design language used to make developing website easier. It is basically used to define the styles of webpages, including their layout, design, variations in display for various devices and sizes of screens [38].

CHAPTER FOUR: DESIGN

4.1 Introduction:

Modeling is widely used in science and engineering to provide abstractions of a system at some level of precision and detail. The model is then analyzed in order to obtain a better understanding of the system being developed. According to the Object Modeling Group (OMG), modeling is the designing of software applications before coding.

In model based software design and development, software modeling is used as an essential part of the software development process. Models are built and analyzed prior to the implementation of the system, and are used to direct the subsequent implementation.

A graphical modeling language such as UML helps in developing, understanding, and communicating the different views [39].

In this chapter, I will use some of UML diagrams to design and describe the developed application in order to achieve good implementation and provide fluent description to the system.

4.2 Design Requirements:

4.2.1 Modeling Language:

Models are needed in system development because they help to understand the interesting characteristics of an existing or desired (complex) system and its environment. They allow engineers to communicate their understanding and design intent to others [40].

Software model is an engineering model of a software system from one or more viewpoints specified using one or more modeling languages. Modeling language is a computer language intended for constructing models of software programs and the contexts in which they operate [40].

4.2.2 Unified Modeling Language:

4.2.2.1 Definition:

One of the most prominent modeling languages used in software design is the unified modeling language (UML). UML is one of the most exciting and useful tools in the world of system development that provides a standardized mechanism to model software systems and even its requirements [41] [42]. It is a visual modeling language that enables system builders to create blueprints that capture their visions in a standard, easy to understand way, and provides a mechanism to effectively share and communicate these visions with others [41].

The value of UML based models comes from their ability to facilitate communication, discussion, documentation for large and complex software systems [42].

4.2.2.2 Types of UML Diagrams:

The nature (types) of the UML diagrams can be classified as follows:

- **Structural versus behavioral:** The structural aspect of a diagram illustrates the way a system is organized, whereas the behavioral aspect models the flow of the system. Structure, for example, shows how classes relate to each other in a class diagram. Behavior shows the way in which a user interacts with the system through a use case or an activity diagram, for example [42].
- **Static versus dynamic**: The static versus dynamic aspect depicts the time dependency of the model. A diagram with no concept of time or movement is static, whereas one that shows changes in time (or even a snapshot in time) is considered dynamic [42].

4.3 Modeling:

4.3.1 System Actors:

An actor is an entity located outside the system that is involved in the interaction with the system described in a use case. It defines a role played by the interactor with the system. An actor may be a human user, external hardware or another system. The standard graphical representation of the actor in UML is the icon called stick man with the name of the actor below the drawing [43] [44].

The system to be developed has two types of actors: admin and user.

4.3.1.1 Admin:

The admin has the right to see sensitive information about the store and accounts. Additionally, he can interact with the database and change the content of the application. The admin can be the MMD head, MD responsible or the storekeeper.

4.3.1.2 User:

The user is the one who have the right to see only the products available in the store with their stock value.

4.3.2 Use Cases:

A use case describes a set of activities and scenarios of a system from the point of view of its actors which lead to a useful outcome for the actors. It is always an initiated by actor [41] [43]. After interviewing with the storekeeper, I have defined the following use cases.

- Authentication.
- DisplayCategories.
- ManageCategories.
- DisplayStockType.
- ManageStockType.
- DisplayProducts.
- MangeProducts.
- DisplayStockFlow.
- ManageFlow
- ProductStockLog
- ManageAdminsUsers.
- AddCategory.
- UpdateCategory.
- AddStockType.
- UpdateStocktype.
- AddProduct.
- UpdateProduct.
- AddFlow.
- Edit.
- AddUser.
- UpdateAccount.
- Delete.
- Sort.
- Filter
- Logout.
- Products.

4.3.3 Use Case Diagrams:

Use Case Diagram: is a model of the requirements of a system at a high level. It visualize the interaction between a set of use cases and the actors involved in these use cases. The diagram itself is not a use case but rather a visual of actors and a group of related use cases [43] [42].

Functional Requirements: are features of the system required by the user. They are determined through an interview with the user. The use cases describe the requirements provided by the system to the user.

Actors: interact with the system to get a useful outcome. In the use case diagram, the actor is represented by a stick man icon.

Relationships: are the relations between an actor and another actor, an actor and a use cases or a use case and another use case. They can be as follow:

- **Actor to Actor Relationship:** This is the generalization relationship and represented by the inheritance arrow [42]. Generalization allows sub use cases to inherit behavior and semantics from super use cases. In other words, sub use cases can partially overwrite the inherited behavior and add further behavior [43].
- **Actor to Use Case Relationship:** is called an association, also occasionally called "communication" because it represents a communication between the actor and the system. This association is represented by a straight line. The association is an interface through which an actor interacts with the system [42].
- **Use Case Relationships:** are three relationships: inherit, extend, and include. They are described below.
    - **Inherit:** Also known as generalization, literally means a use case implements the behavior described by another higher-level abstract use case. While an abstract use case contains a generalized description of how an actor will use the system, the corresponding concrete use

case, which is inherited from the generalized use case, will describe the actual steps in the identification of the user.

○ **Include:** When a part of the behavior documented within a use case is likely to be reused by other parts of a system, it is advisable to factor out that common behavior and show it as an independent use case. This newly created use case can then be “included” in the original use case from which it was factored out. The newly created use case also becomes available to other use cases in the system. The relationship between two such use cases is that of includes, and the arrowhead on this relationship points from the including use case to the use case being included.

○ **Extend:** When a use case extends or specializes the behavior of another use case, the relationship is “extends”. This extension may be undertaken in order to add functionality to an existing use case. The extension can also superimpose a new special type of behavior on an existing behavior. The extension relationship may also describe an anomalous situation or an exception, beyond that provided in the base use case. The extending use case may thus be thought of as representing an “option”. In the “extends” relationship between two use cases, the arrowhead points to the use case that is being extended [42].

Table 1 determines the arrow used for each relationship described above.

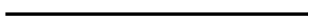
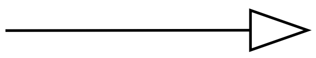
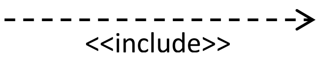
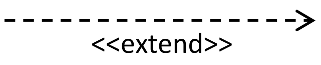
Relationship	**Used Arrow.**
Association	
Inherit	
Include	
Extend	

Table 1: Arrows Used in Use Case Diagram’s Relationships

4.3.3.1 Admin Use Case Diagram:

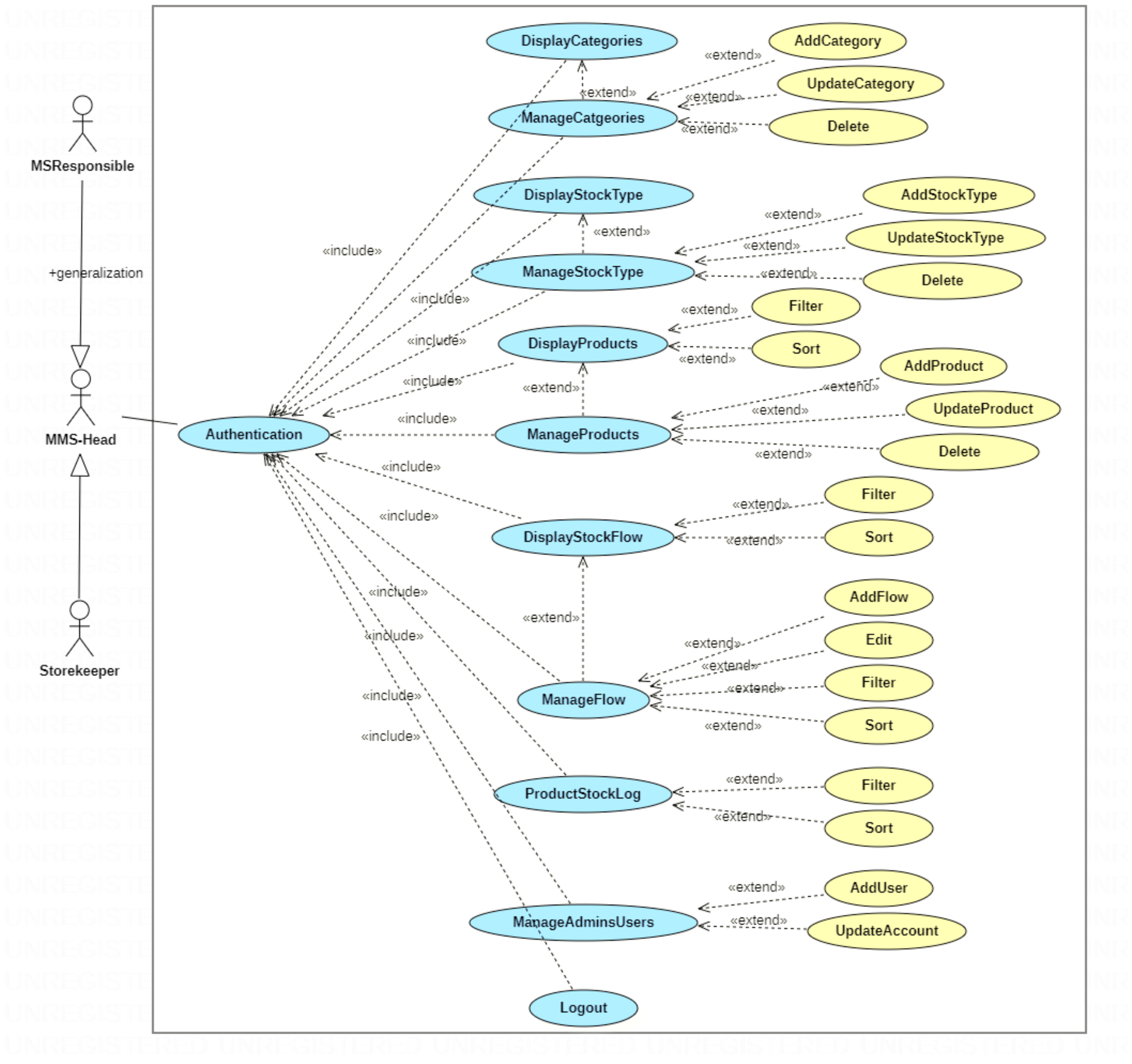


Figure 4: Admin UCD.

4.3.3.2 User Use Case Diagram:

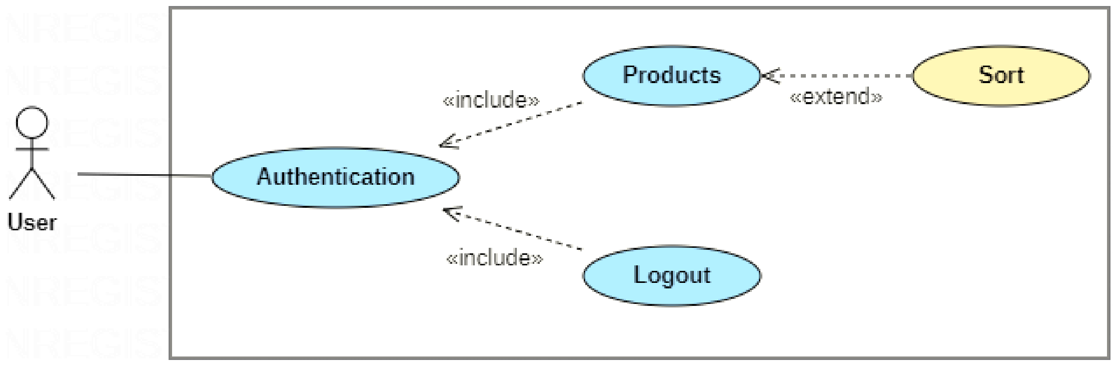


Figure 5: User UCD.

4.3.4 Textual Description of Use Cases:

Authentication:

Name: Authentication
Description: This use case allows the admin and the user to access their corresponding area of the application.
Actors: Admin (The MMD Head, the MD Responsible and the storekeeper) and user.
Preconditions: Browser and an internet connection.
Scenario:
1- Open Browser.
2- The actor starts the application.
3- The system checks authentication.
4- If the actor is not authenticated, it will ask for login.
5- The actor enters the required username and password.
6- The system checks username and password.
7- If account details are correct, the system allows the actor to access the application.
Result: Actor accessed the application.
Alternative Flow: **-** If the actor is authenticated, access directly the application.
- After 5 incorrect logins, System disables login for 10 minutes.

Figure 6: Authentication Use Case Description.

Logout Use Case:

Name: Logout
Description: This use case allows the actor to logout from the application.
Actors: Admin (The MMD Head, the MD Responsible and the storekeeper) and User.
Preconditions: Actor is already accessing the application.
Scenario:
1- The actor clicks on the logout tool.
2- The system delete authentication of the actor and end its session.
3- The user loss access to the application.
Post-conditions: The application waits for authentication.
Result: Actor losses access to the application.

Figure 7: Logout Use Case Description.

DisplayCategories Use Case:

Name: DisplayCategories

Description: This use case allows the actor of type admin to see the available categories of products in the store.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Categories tool.

2- The system retrieve the data from the database.

3- The system displays the content to the actor.

Result: Actor see the available categories.

Figure 8: DisplayCategories Use Case Description.

ManageCategories Use Case:

Name: ManageCategories.

Description: This use case allows the actor of type admin to see the available categories of products in the store with extra tools. These tools are: add, update, and delete.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Manage Categories tool.

2- The system retrieve the data from the database.

3- The system displays the content to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: Actor have access to categories management tools.

Figure 9: ManageCategories Use Case Description.

DisplayStockType Use Case:

Name: DisplayStockType.

Description: This use case allows the actor to see the available stock type.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Stock Type tool.

2- The system retrieve the data from the database.

3- The system displays the content to the actor.

Result: Actor see the available stock type.

Figure 10: DisplayStockType Use Case Description.

ManageStockType Use Case:

Name: ManageStockType.

Description: This use case allows the actor to see the available stock type of products in the store with extra tools. These tools are: add, update, and delete stock type.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Manage Stock Type tool.

2- The system retrieve the data from the database.

3- The system displays the content provided with extra tools to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: Actor have access to stock type management tools.

Figure 11: ManageStockType Use Case Description.

DisplayProducts Use Case:

Name: DisplayProducts.

Description: This use case allows the actor to see the available products in the store.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Products tool.

2- The system retrieve the data from the database.

3- The system displays the content to the actor.

Result: Actor see the available products.

Figure 12: DisplayProducts Use Case Description.

ManageProducts Use Case:

Name: ManageProducts.

Description: This use case allows the actor to see the available products in the store. Extra tools are provided to help him in adding, updating, deleting a product, filtering and sorting the content.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Manage Products tool.

2- The system retrieve the data from the database.

3- The system displays the content provided with extra tools to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: Actor have access to products management tools.

Figure 13: ManageProducts Use Case Description.

DisplayStockFlow Use Case:

Name: DisplayStockFlow.

Description: This use case allows the actor to see the previous made transactions of the store's products. Extra tools are provided to help him in filtering and sorting the content.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Stock Flow tool.

2- The system retrieve the data from the database.

3- The system displays the required content to the actor.

Result: Actor see the history of all made transaction in the store.

Figure 14: DisplayStockFlow Use Case Description.

ManageFlow Use Case:

Name: ManageFlow.

Description: This use case allows the actor to see the flow of stock. Extra tools are provided to help him in flow management.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Manage Flow tool.

2- The system retrieve the data from the database.

3- The system displays the content provided with extra tools to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: Actor have access to products management tools.

Figure 15: ManageFlow Use Case Description.

<u>ProductStockLog Use Case:</u>

Name: ProductStockLog.

Description: This use case allows the actor to see the previous made transactions specific product. Extra tools are provided to help him in filtering and sorting the content.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Product Log tool.

2- The actor select the category of the product.

3- The actor select the product and submit the selected choices.

4- The system retrieve the data from the database.

5- The system displays the required content to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: The actor sees the log of the stock of the product.

Figure 16: ProductStockLog Use Case Description.

ManageAdminsUsers Use Case:

Name: ManageAdminsUsers.

Description: This use case allows the actor to see the available users and admins accounts. Extra tools are provided to help him in management.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on the Admin tool.

2- The system retrieves the data from the database.

3- The system displays the required content to the actor.

Post-condition: The app waits to respond to the actor in case he uses one of the extra tools.

Result: The actor sees the available accounts with extra tools provided.

Figure 17: ManageAdminsUsers Use Case Description.

AddCategory Use Case:

Name: AddCategory.

Description: This use case allows the actor to add a category.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Categories interface.

Scenario:

1- The actor fills the name field.

2- The actor submit information

3- The system Process the information and creates a category.

4- The system changes the interface to Categories interface.

5- The system displays a success message.

Result: The actor added a category.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 18: AddCategory Use Case Description.

UpdateCategory Use Case:

Name: UpdateCategory.

Description: This use case allows the actor to update an existing category.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Categories interface.

Scenario:

1- The actor edit the name.

2- The actor submit information.

3- The system Process the information and updates the category.

4- The system changes the interface to Categories interface.

5- The system displays a success message.

Result: The actor updated a category.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 19: UpdateCategory Use Case Description.

AddStockType Use Case:

Name: AddStockType.

Description: This use case allows the actor to add a stock type.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage stock type interface.

Scenario:

1- The actor fills the name field.

2- The actor submit information

3- The system Process the information and creates a stock type.

4- The system changes the interface to Stock Type interface.

5- The system displays a success message.

Result: The actor added a stock type.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 20: AddStockType Use Case Description.

UpdateStockType Use Case:

Name: UpdateStockType.

Description: This use case allows the actor to update an existing stock type.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Stock Type interface.

Scenario:

1- The actor edit the name.

2- The actor submit information.

3- The system Process the information and updates the stock type.

4- The system changes the interface to Stock Type interface.

5- The system displays a success message.

Result: The actor updated a stock type.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 21: UpdateStockType Use Case Description.

AddProduct Use Case:

Name: AddProduct.

Description: This use case allows the actor to add a product.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage products interface.

Scenario:

1- The actor fills the characteristics of the product.

2- The actor submit information

3- The system Process the information and creates a product.

4- The system changes the interface to products interface.

5- The system displays a success message.

Result: The actor added a product.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 22: AddProduct Use Case Description.

UpdateProduct Use Case:

Name: UpdateProduct.

Description: This use case allows the actor to update an existing product.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Products interface.

Scenario:

1- The actor edit the product details.

2- The actor submit information.

3- The system Process the information and updates the product.

4- The system changes the interface to Products interface.

5- The system displays a success message.

Result: The actor updated a product.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 23: UpdateProduct Use Case Description.

AddFlow Use Case:

Name: AddFlow.

Description: This use case allows the actor to add a stock flow.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Flow interface.

Scenario:

1- The actor fills the characteristics of the flow (transaction).

2- The actor submit information

3- The system Process the information and records a flow.

4- The system displays a success message.

Result: The actor recorded a flow.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 24: AddFlow Use Case Description.

Edit Use Case:

Name: Edit.

Description: This use case allows the actor to correct mistakes in the last made transaction of a specific product.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Manage Flow interface.

Scenario:

1- The actor corrects the mistakes of the flow.

2- The actor submit information

3- The system Process the information and edit the flow.

4- The system changes the interface to Manage Flow interface.

5- The system displays a success message.

Result: The actor edited a flow.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 25: Edit Use Case Description.

AddUser Use Case:

Name: AddUser.

Description: This use case allows the actor to add a user account.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Admin interface.

Scenario:

1- The actor fills the account details with his password included for security purpose.

2- The actor submit information.

3- The system checks the admin password.

4- If the admin password is correct, the system Process the information and creates an account.

5- The system displays a success message.

Result: The actor created a user account.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 26: AddUser Use Case Description.

UpdateAccount Use Case:

Name: UpdateAccount.

Description: This use case allows the actor to update an account.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to Admin interface.

Scenario:

1- The actor edit the account details as desired with admin password included for security.

2- The actor submit information.

3- The system checks the admin password.

4- If the admin password is correct, the system Process the information and updates the account.

5- The system displays a success message.

Result: The actor updated the account.

Alternative Flow: The system displays an errors or validation errors message in case process failed.

Figure 27: UpdateAccount Use Case Description.

Products Use Case:

Name: Products.

Description: This use case allows the actor to see the available products in the store with their stock.

Actors: User.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on Product tool.

2- The system retrieve the data from the database.

3- The system displays the content to the actor.

Post-condition: The app waits to respond in case the user use the sort tool.

Result: Actor sees the available products.

Figure 28 DisplayProducts Use Case Description.

Delete Use Case:

Name: Delete.

Description: This use case allows the actor to delete category, stock type, product or account.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor clicks on the delete button.

2-The system displays a confirmation window.

3- If the actor confirms deletion, the system perform deletion.

5- The system displays a success message.

Result: The actor perform deletion.

Alternative Flow: Deletion canceled. The system waits to respond to the actor in case he uses one of the extra tools.

Figure 29: UpdateAccount Use Case Description.

Filter Use Case:

Name: Filter.

Description: This use case allows the actor to filter the content of an interface.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor selects an option of the filter.

2- The actor then selects the value(s) related to the selected filter's option.

3- The actor submit information.

4- The system process the submission and filter the content.

Post-condition: The system waits to respond to the actor in case he uses one of the extra tools.

Result: The actor get the content filtered.

Figure 30: Filter Use Case Description.

Sort Use Case:

Name: Sort.

Description: This use case allows the actor to sort the content of an interface.

Actors: The MMD Head, the MD Responsible and the storekeeper.

Preconditions: Access to the application.

Scenario:

1- The actor selects an option of the sort tool.

2- The actor submit information.

3- The system process the submission and sort the content.

Post-condition: The system waits to respond to the actor in case he uses one of the extra tools.

Result: The actor get the content sorted.

Figure 31: Sort Use Case Description.

CHAPTER FIVE: IMPLEMENTATION

5.1 Introduction:

A design process is followed by the implementation process. This chapter introduces the Implementation stage of the project, and the description of the developed application including its interface, features, tools and functions.

5.2 Database:

Any web application requires a database that will use to Store, retrieve, update and even delete the data. This section shows the implementation of the database of the application.

5.2.1 Database Schema:

The database created for the web application is igee_gms. The following schema diagram [45] shown in Figure 29 describes the relations of the database.

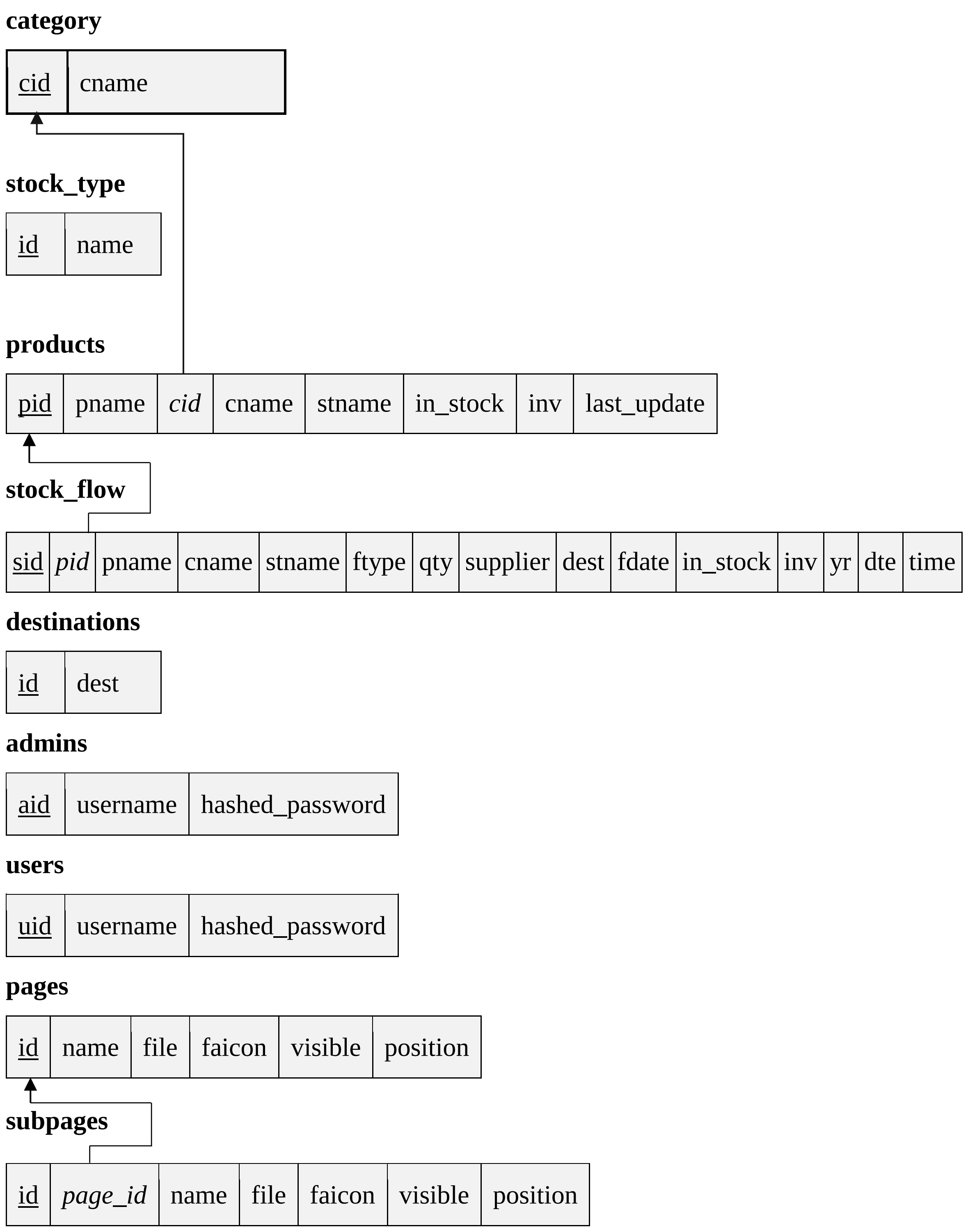
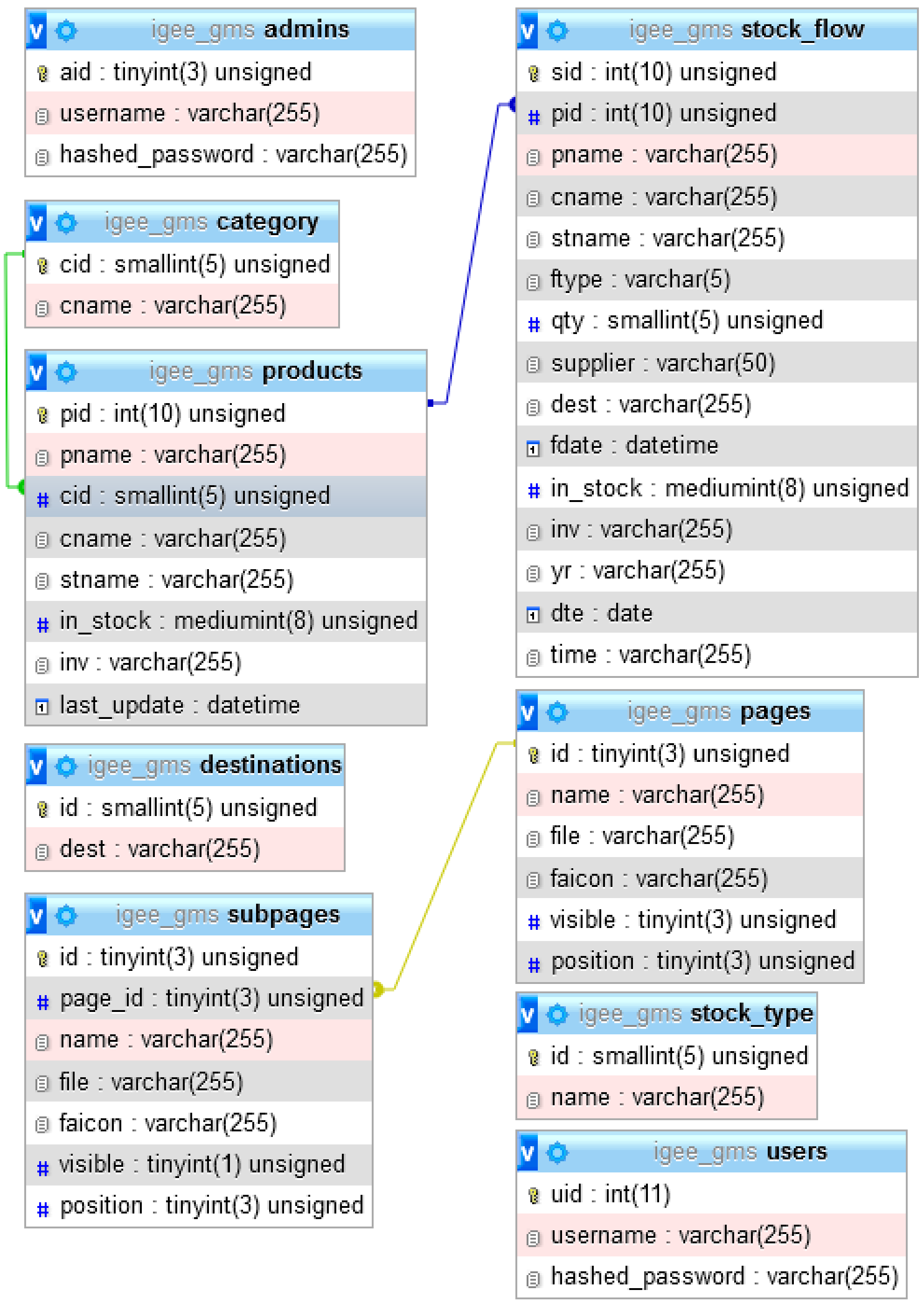

category
cid | cname
stock_type
id | name
products
pid | pname | cid | cname | stname | in_stock | inv | last_update
stock_flow
sid | pid | pname | cname | stname | ftype | qty | supplier | dest | fdate | in_stock | inv | yr | dte | time
destinations
id | dest
admins
aid | username | hashed_password
users
uid | username | hashed_password
pages
id | name | file | faicon | visible | position
subpages
id | page_id | name | file | faicon | visible | position

Figure 32 Schema Diagram of the Database igee_gms

Primary Keys are underlined and foreign keys are the columns from which the arrow start and they are italic.

5.2.2 Database Relations and Their relationships:

The following Database design shown in Figure 30 describe the structure of the database and its relations. Besides, it shows the relationships between relations. These relationships are one to many relationships.



igee_gms **admins**
- aid : tinyint(3) unsigned
- username : varchar(255)
- hashed_password : varchar(255)

igee_gms **category**
- cid : smallint(5) unsigned
- cname : varchar(255)

igee_gms **products**
- pid : int(10) unsigned
- pname : varchar(255)
- cid : smallint(5) unsigned
- cname : varchar(255)
- stname : varchar(255)
- in_stock : mediumint(8) unsigned
- inv : varchar(255)
- last_update : datetime

igee_gms **destinations**
- id : smallint(5) unsigned
- dest : varchar(255)

igee_gms **subpages**
- id : tinyint(3) unsigned
- page_id : tinyint(3) unsigned
- name : varchar(255)
- file : varchar(255)
- faicon : varchar(255)
- visible : tinyint(1) unsigned
- position : tinyint(3) unsigned

igee_gms **stock_flow**
- sid : int(10) unsigned
- pid : int(10) unsigned
- pname : varchar(255)
- cname : varchar(255)
- stname : varchar(255)
- ftype : varchar(5)
- qty : smallint(5) unsigned
- supplier : varchar(50)
- dest : varchar(255)
- fdate : datetime
- in_stock : mediumint(8) unsigned
- inv : varchar(255)
- yr : varchar(255)
- dte : date
- time : varchar(255)

igee_gms **pages**
- id : tinyint(3) unsigned
- name : varchar(255)
- file : varchar(255)
- faicon : varchar(255)
- visible : tinyint(3) unsigned
- position : tinyint(3) unsigned

igee_gms **stock_type**
- id : smallint(5) unsigned
- name : varchar(255)

igee_gms **users**
- uid : int(11)
- username : varchar(255)
- hashed_password : varchar(255)

Figure 33 Database Relations and Relationships

5.2.3 Data Description:

The following table explain the meaning of each field or attribute of a corresponding relation.

category	
cid	Category ID
cname	Category name
Stock_type	
id	Stock type ID
name	Stock type
products	
pid	Product ID
pname	Product name
cid	Category ID
cname	Category name
stname	Stock type
in_stock	Available quantity of product in store
inv	The inventory number
last_update	Last time at which the product is delivered
stock_flow	
sid	Transaction (flow) ID.
pid	Product ID.
pname	Product name.
cname	Category name.
stname	Stock type.
ftype	Flow (transaction) type. Bought or supply.
qty	Quantity purchased or delivered.
dest	Institute Department which order the product.
fdate	The date and time at which transaction happens.
in_stock	Available quantity of product in store.
inv	The inventory number.
yr	The year part from fdate.
dte	The date part from fdate.
time	The time part from fdate.
destinations	
id	The Destination ID.
dest	Name of the destination of the product.

admins	
aid	The admin account ID.
username	The name of the admin account.
hashed_password	The encrypted password of the admin account.
users	
uid	The user account ID.
username	The name of the user account.
hashed_password	The encrypted password of the user account.
pages	
id	The page ID.
name	The name of the page.
file	The name of the PHP file of the page.
faicon	Font Awesome Icons: The i tag of the icon of buttons of the application.
visible	Determines the page privacy if it is public (user) or private (admin).
position	Helps in ordering page when retrieved from the database.
subpages	
id	The ID of the subpage of the main page.
page_id	The ID of parent page of the subpage.
name	The name of the subpage.
file	The name of the PHP file of the subpage.
faicon	Font Awesome Icons: The i tag of the icon of menu buttons of the application.
visible	Determines the subpage privacy if it is public (user) or private (admin).
position	Helps in ordering pages when retrieved from the database.

Table 2: Data Description

The relations pages and subpages are used by the application in order to create the navigation bar of the website dynamically using PHP.

5.3 The Architecture of the Application:

The following diagram shown in Figure 31 gives a general idea about the application structure and its pages.

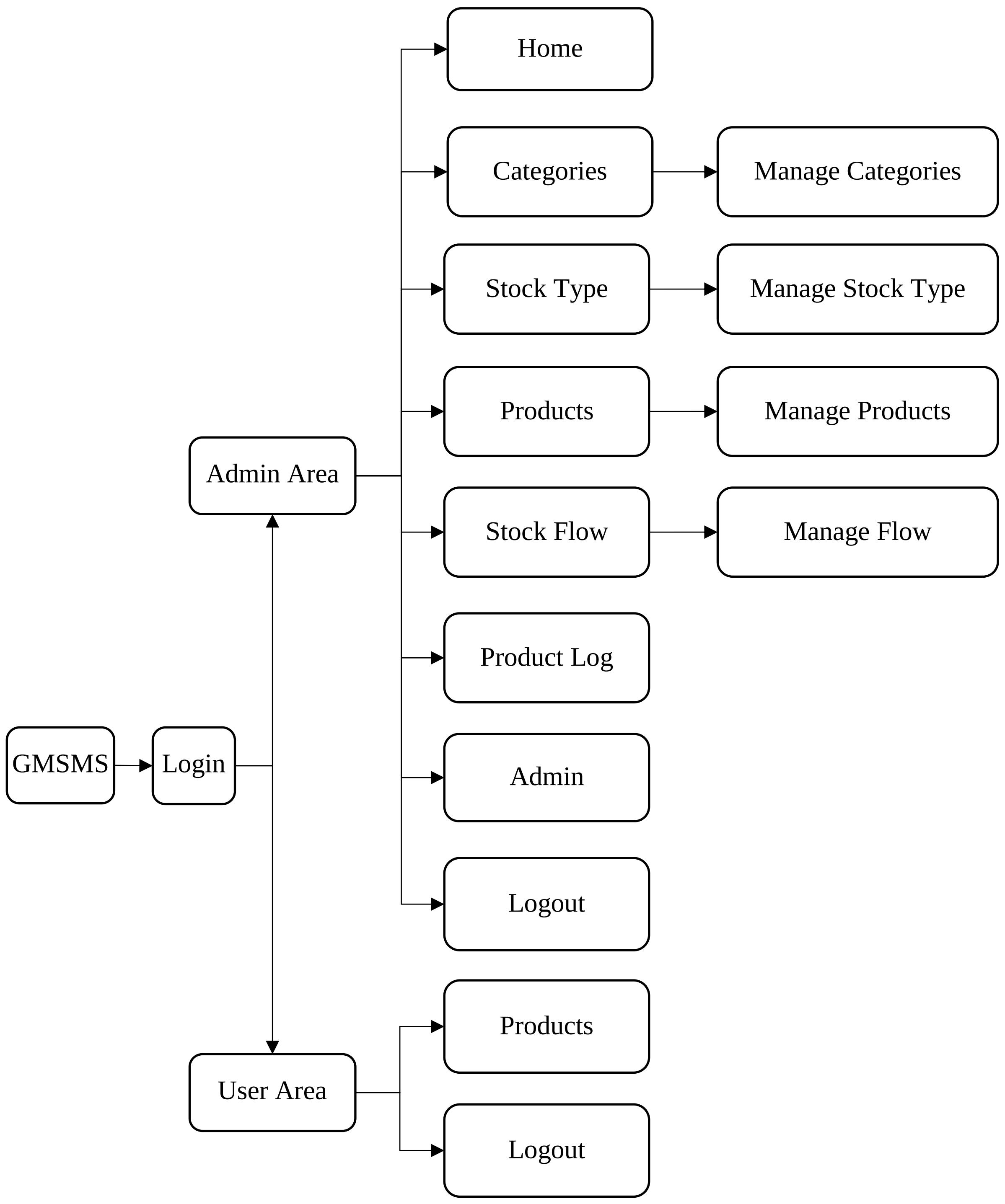


Figure 34 Architecture of the Application

5.4 Interfacing The Application:

The developed application has two area: admin area and semipublic area. The admin area has all privileges of CRUD operations upon the database. Also he can see all details about the store. On the other hand, the semipublic area is not for all public people. It is for public users inside the institute whom the admin decide to allow them to see the products existed in the store.

5.4.1 Login Interface:

Login Page: If the user or the admin are not authenticated. They will have no access to the application. An automatic redirection will redirect any unauthenticated access to the login page. When the process of login succeed, the application will redirect the person who want to access it to the corresponding area based on the type of the authenticated account: if it is a user account, the application will redirect to the user (semipublic) area. Otherwise, it will redirect to the admin area. For incorrect logins, after 5 attempts, the application will prevent the login process for 10 minutes. Figure 32 shows below the interface of the login page.

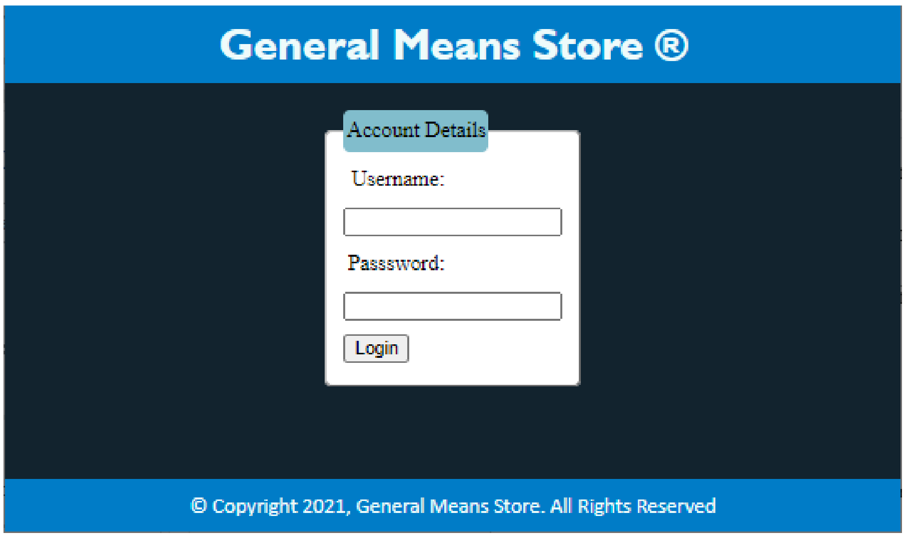
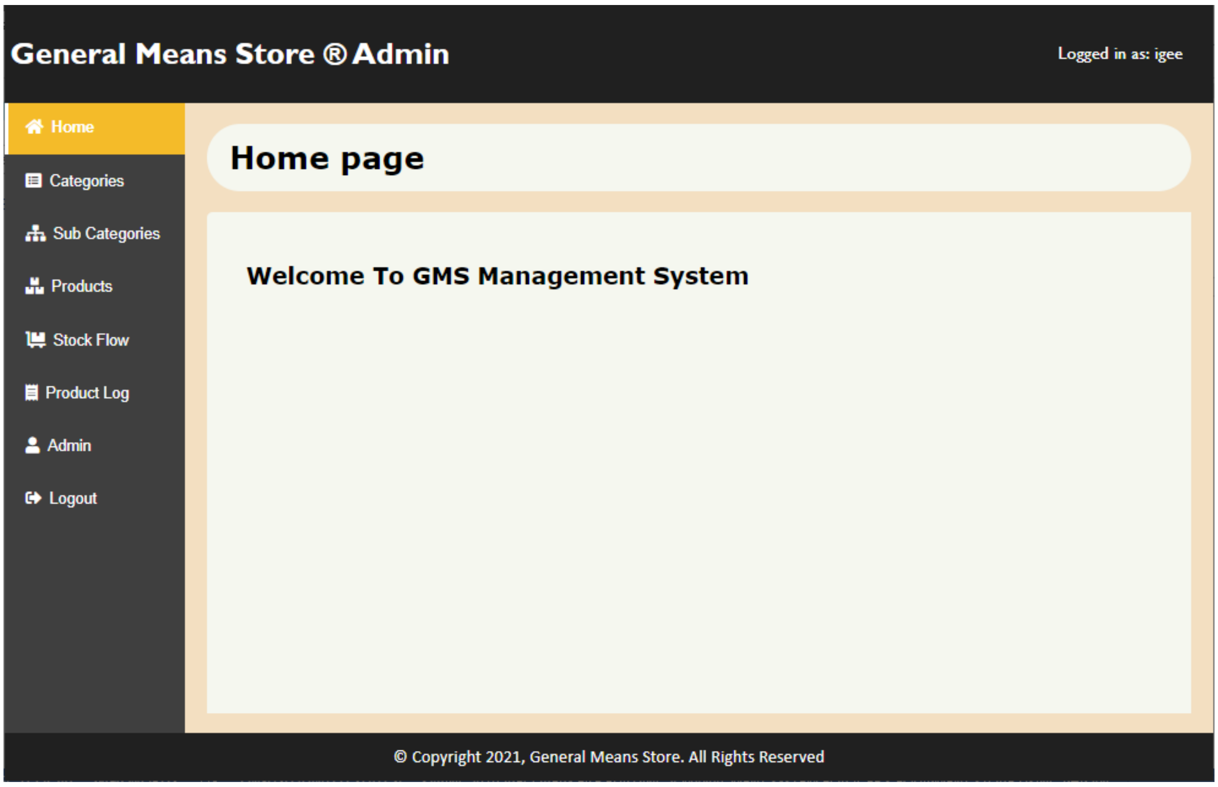


Figure 35: Login Interface

5.4.2 The Admin Area:

The admin area has the following interfaces as described below. The admin area can be accessed only by the MMD head, MD responsible and the storekeeper.

Home: This is the home page of the application. No operations can be done using it. Its interface is shown in Figure 33.



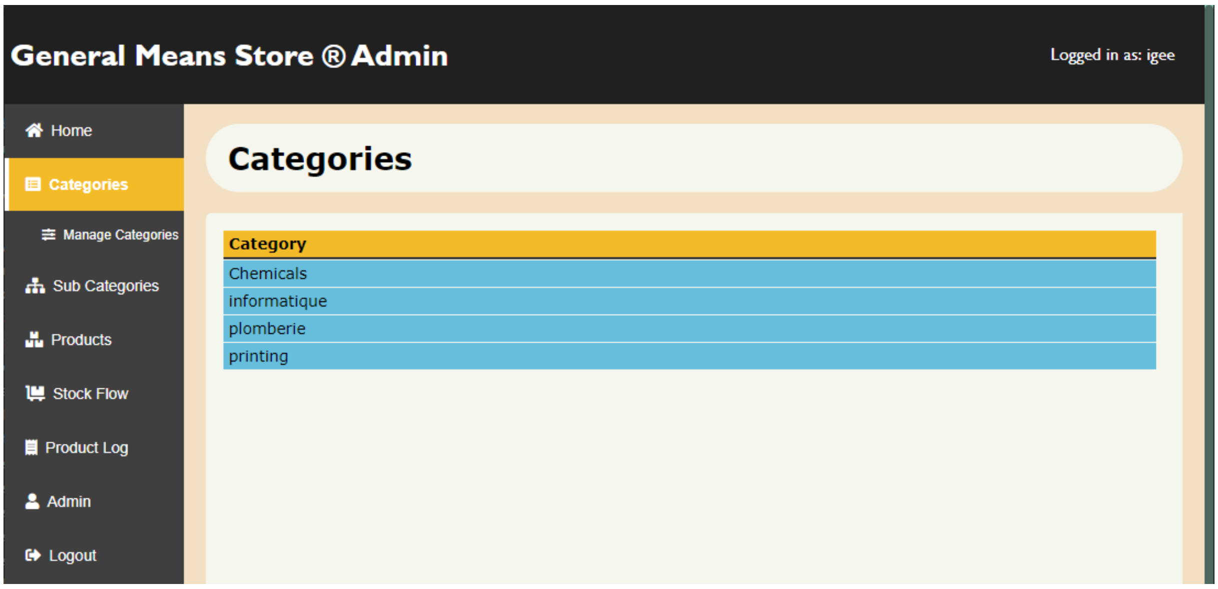
Home page

Welcome To GMS Management System

© Copyright 2021, General Means Store. All Rights Reserved

Figure 36: Home Page Interface

Categories: the categories tool provides to the admin the available categories of products. Figure 34 show the interface of the Categories page.

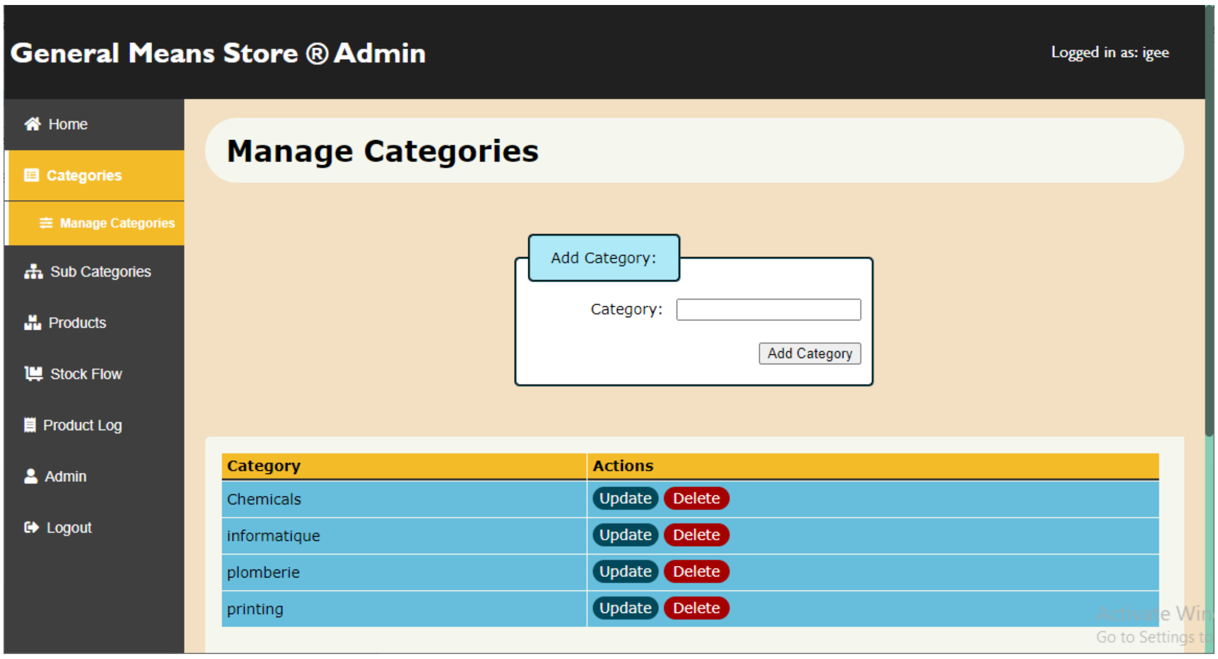


Categories

Category
Chemicals
informatique
plomberie
printing

Figure 37: Categories Interface

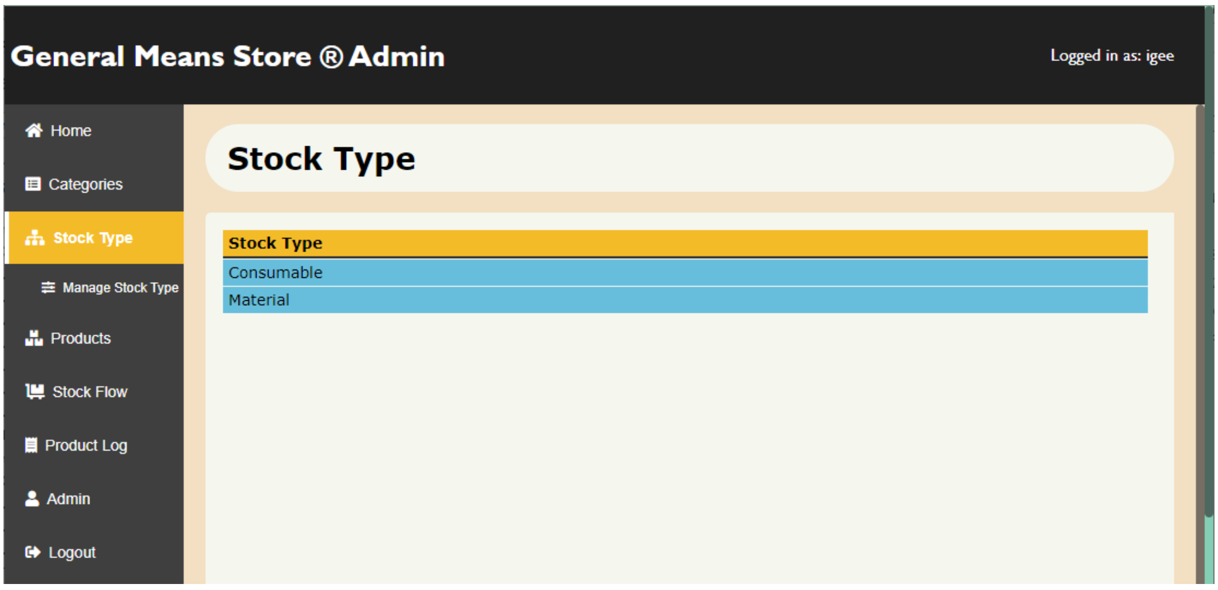
Manage Categories: This tool allows the admin to update or delete the existing categories. In addition, it allows him to create a new category. Figure 35 shows its interface.



Category	Actions
Chemicals	Update Delete
informatique	Update Delete
plomberie	Update Delete
printing	Update Delete

Figure 38: Manage Categories Interface

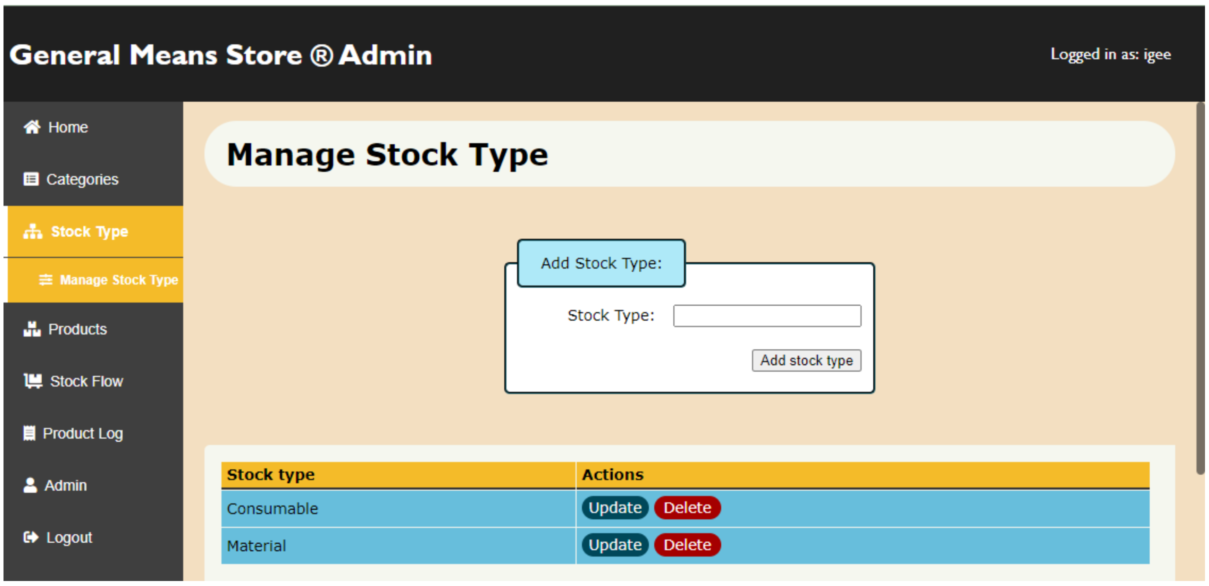
Stock Type: The Stock Type interface shows to the admin the available stock types of products in a table. Figure 36 shows the interface of stock type page.



Stock Type
Consumable
Material

Figure 39: Stock Type Interface

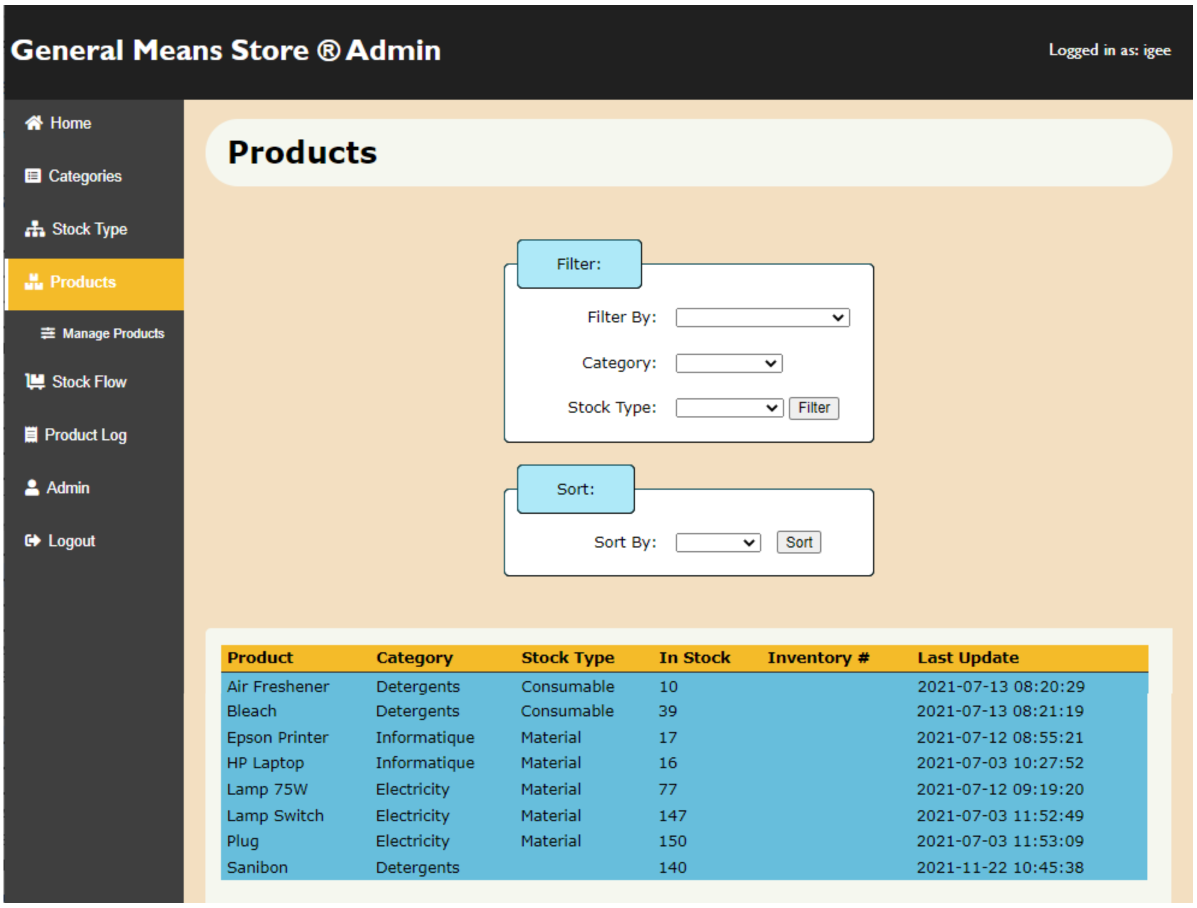
Manage Stock Type: In this page, the admin has the privilege to update or delete the existing stock type. Also, He can create a new stock type. The following figure shows the interface of this page.



Stock type	Actions
Consumable	Update Delete
Material	Update Delete

Figure 40: Manage Stock Type Interface

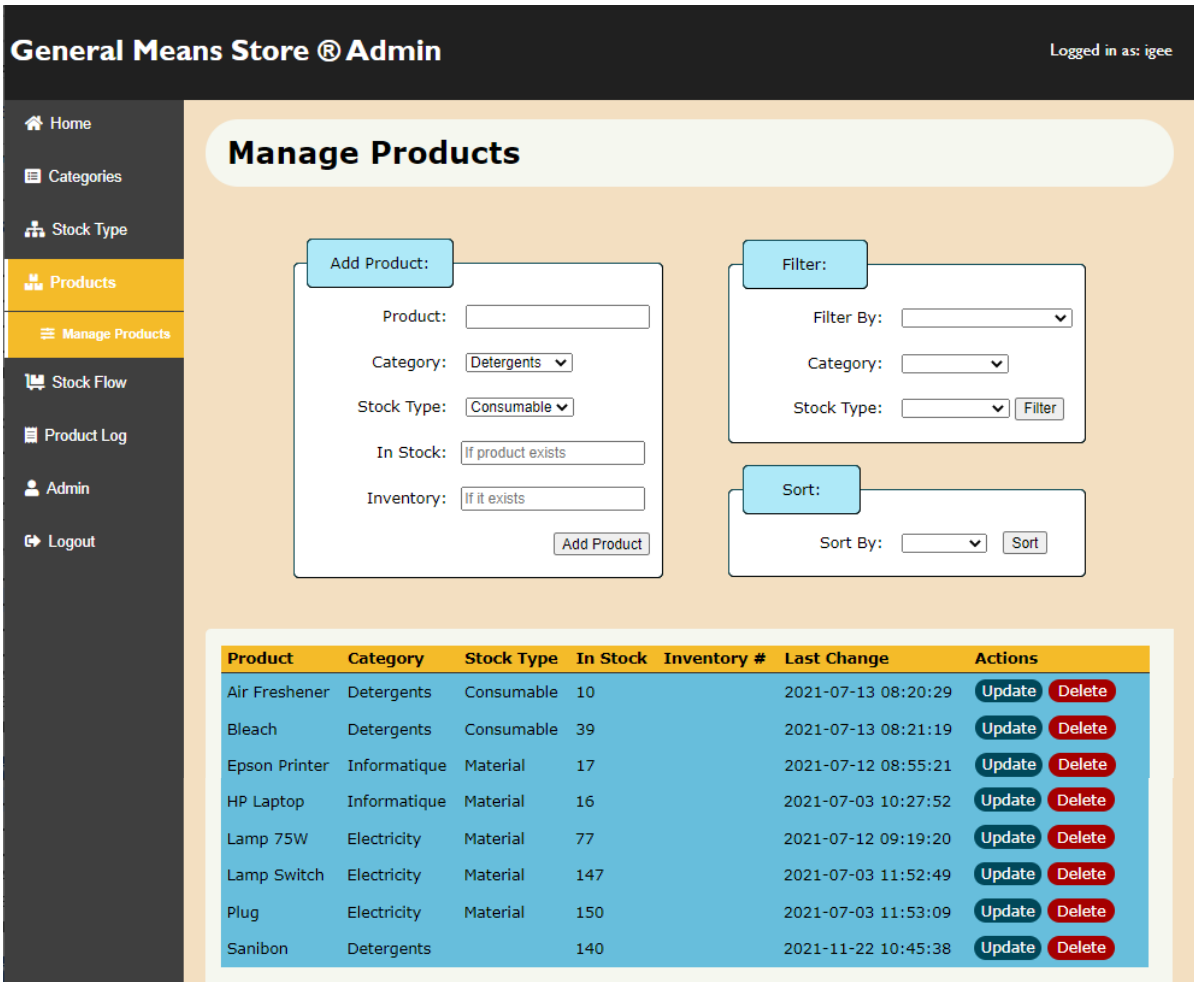
Products: The products tool provides the admin with the existing products in the store as well as with the necessary details. It is provided with tools for filtering and sorting the content of the page which will help him in monitoring the store. The admin has to choose a filter/sort option from the Filter By/Sort list and select or fill the value of the selected option. After that, He need to click on Filter or Sort button to get the sorted/filtered content. Its interface is shown below in Figure 38.



Product	Category	Stock Type	In Stock	Inventory #	Last Update
Air Freshener	Detergents	Consumable	10		2021-07-13 08:20:29
Bleach	Detergents	Consumable	39		2021-07-13 08:21:19
Epson Printer	Informatique	Material	17		2021-07-12 08:55:21
HP Laptop	Informatique	Material	16		2021-07-03 10:27:52
Lamp 75W	Electricity	Material	77		2021-07-12 09:19:20
Lamp Switch	Electricity	Material	147		2021-07-03 11:52:49
Plug	Electricity	Material	150		2021-07-03 11:53:09
Sanibon	Detergents		140		2021-11-22 10:45:38

Figure 41: Products Interface

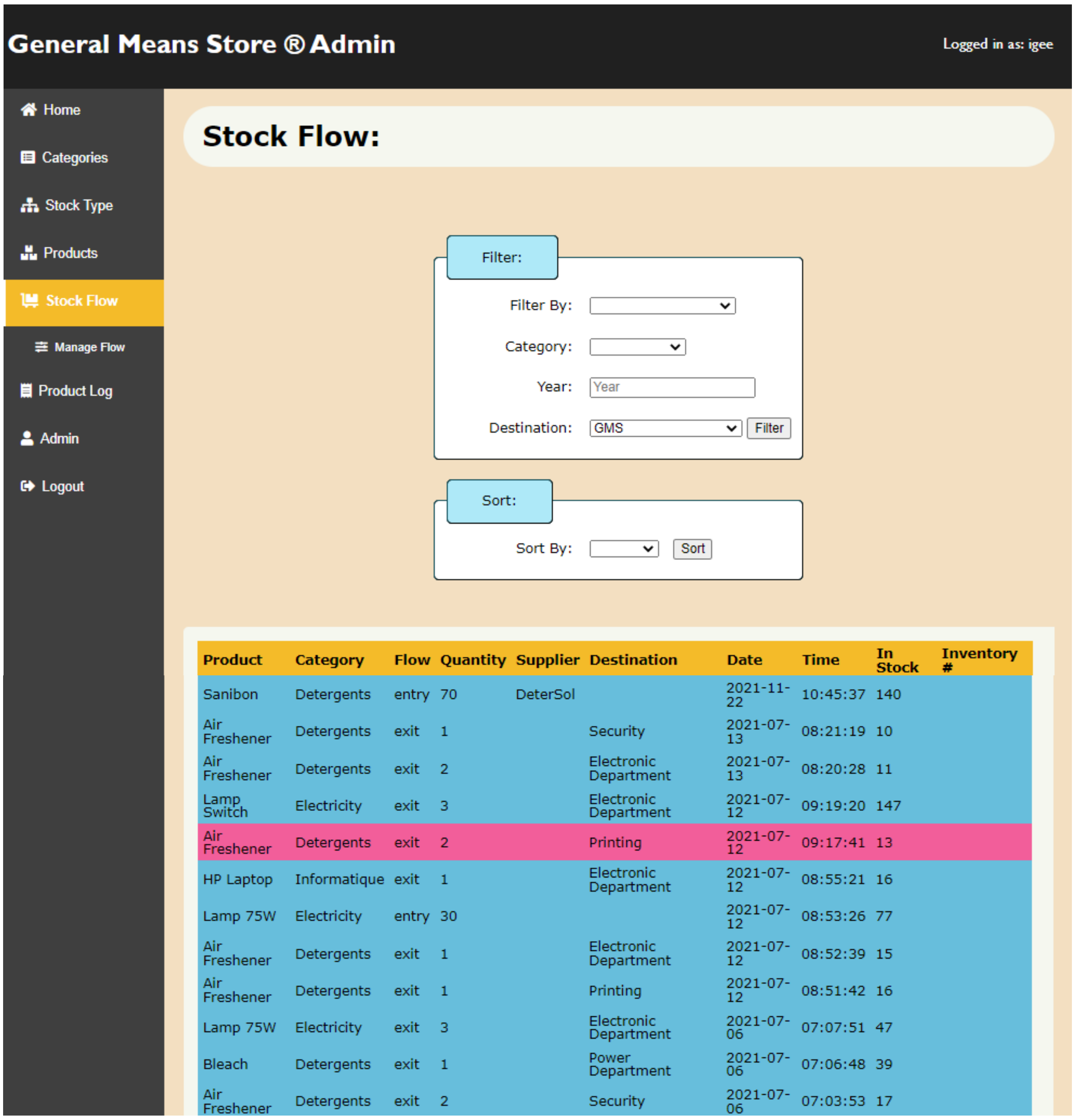
Manage Products: This tool allows the admin to update or delete the existing products as his desire. He can also add new product to the database. In case an error happens in filling the fields or the product exist, the application will notify the user about that. As the interface of products page, the admin has a filtering and sorting tools that helps him in managing and monitoring the store. He can sort and filter the content of the page based on the available filter and sort options. Figure 39 shows the Manage Products interface.



Product	Category	Stock Type	In Stock	Inventory #	Last Change	Actions
Air Freshener	Detergents	Consumable	10		2021-07-13 08:20:29	Update Delete
Bleach	Detergents	Consumable	39		2021-07-13 08:21:19	Update Delete
Epson Printer	Informatique	Material	17		2021-07-12 08:55:21	Update Delete
HP Laptop	Informatique	Material	16		2021-07-03 10:27:52	Update Delete
Lamp 75W	Electricity	Material	77		2021-07-12 09:19:20	Update Delete
Lamp Switch	Electricity	Material	147		2021-07-03 11:52:49	Update Delete
Plug	Electricity	Material	150		2021-07-03 11:53:09	Update Delete
Sanibon	Detergents		140		2021-11-22 10:45:38	Update Delete

Figure 42: Manage Products Interface

Stock Flow: This tool displays to the admin the log of transactions with all necessary details. He can also filter and sort the content of the page. Figure 40 shows its interface.



Stock Flow:

Filter:

Filter By:

Category:

Year: Year

Destination: GMS

Filter

Sort:

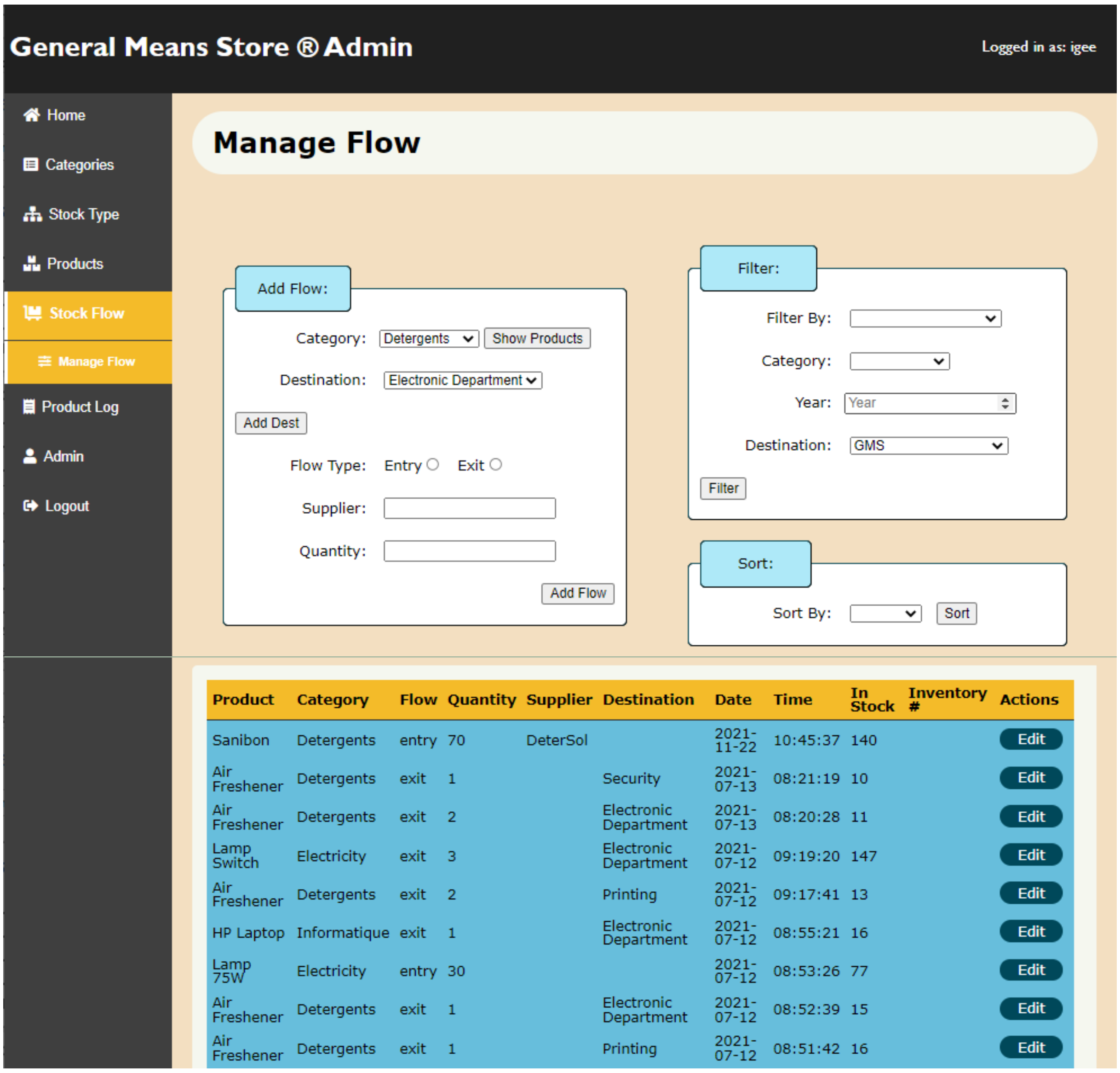
Sort By:

Sort

Product	Category	Flow	Quantity	Supplier	Destination	Date	Time	In Stock	Inventory #
Sanibon	Detergents	entry	70	DeterSol		2021-11-22	10:45:37	140	
Air Freshener	Detergents	exit	1		Security	2021-07-13	08:21:19	10	
Air Freshener	Detergents	exit	2		Electronic Department	2021-07-13	08:20:28	11	
Lamp Switch	Electricity	exit	3		Electronic Department	2021-07-12	09:19:20	147	
Air Freshener	Detergents	exit	2		Printing	2021-07-12	09:17:41	13	
HP Laptop	Informatique	exit	1		Electronic Department	2021-07-12	08:55:21	16	
Lamp 75W	Electricity	entry	30			2021-07-12	08:53:26	77	
Air Freshener	Detergents	exit	1		Electronic Department	2021-07-12	08:52:39	15	
Air Freshener	Detergents	exit	1		Printing	2021-07-12	08:51:42	16	
Lamp 75W	Electricity	exit	3		Electronic Department	2021-07-06	07:07:51	47	
Bleach	Detergents	exit	1		Power Department	2021-07-06	07:06:48	39	
Air Freshener	Detergents	exit	2		Security	2021-07-06	07:03:53	17	

Figure 43: Stock Flow Interface

Manage Flow: As stock flow interface, this tools displays the previous performed transactions with all necessary details. However, the admin can record the transaction if an item is being stored or delivered to a department. In addition, the admin can edit a flow if a mistake happened. The page also provides filtering and sorting tools to filter and sort the content of the page. The following figure gives its interface.



Manage Flow

Add Flow:

Category: Detergents | Show Products

Destination: Electronic Department

Add Dest

Flow Type: Entry ○ Exit ○

Supplier:

Quantity:

Add Flow

Filter:

Filter By:

Category:

Year: Year

Destination: GMS

Filter

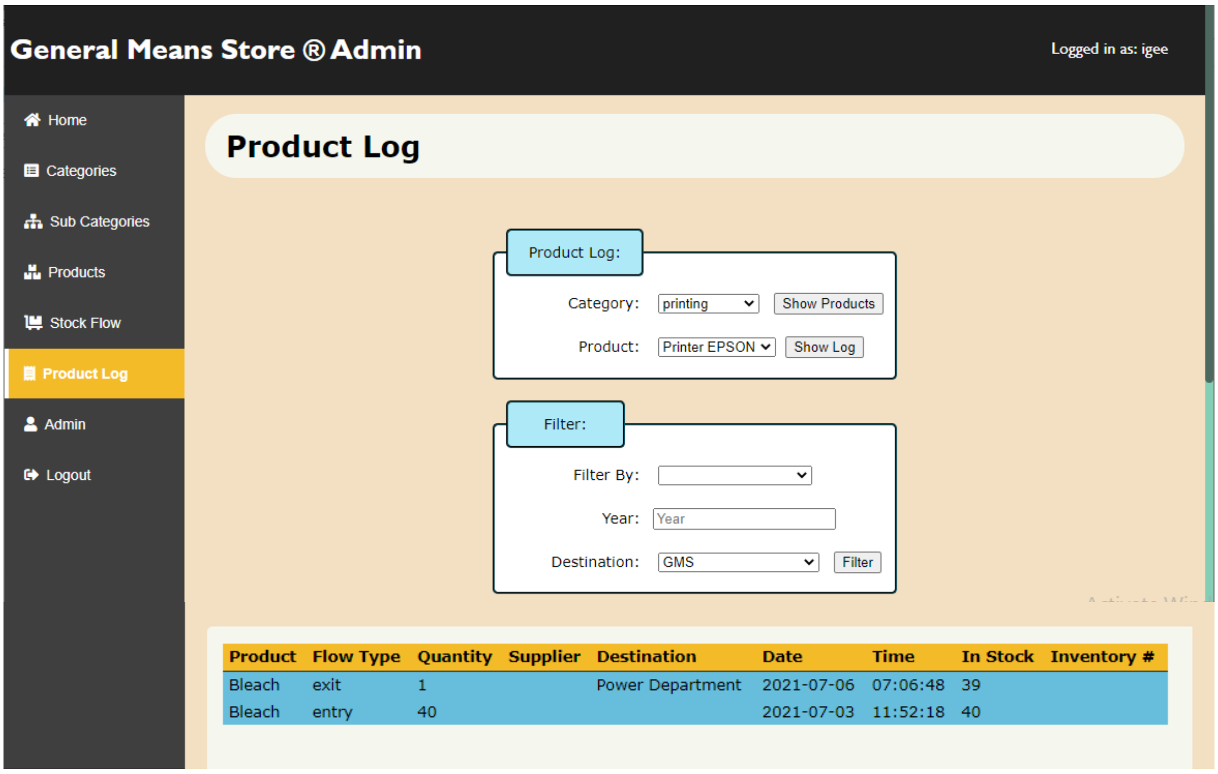
Sort:

Sort By: | Sort

Product	Category	Flow	Quantity	Supplier	Destination	Date	Time	In Stock	Inventory #	Actions
Sanibon	Detergents	entry	70	DeterSol		2021-11-22	10:45:37	140		Edit
Air Freshener	Detergents	exit	1		Security	2021-07-13	08:21:19	10		Edit
Air Freshener	Detergents	exit	2		Electronic Department	2021-07-13	08:20:28	11		Edit
Lamp Switch	Electricity	exit	3		Electronic Department	2021-07-12	09:19:20	147		Edit
Air Freshener	Detergents	exit	2		Printing	2021-07-12	09:17:41	13		Edit
HP Laptop	Informatique	exit	1		Electronic Department	2021-07-12	08:55:21	16		Edit
Lamp 75W	Electricity	entry	30			2021-07-12	08:53:26	77		Edit
Air Freshener	Detergents	exit	1		Electronic Department	2021-07-12	08:52:39	15		Edit
Air Freshener	Detergents	exit	1		Printing	2021-07-12	08:51:42	16		Edit

Figure 44: Manage Flow Interface

Product Log: In this page the admin has the ability to see the product log which is the previous transactions of a certain product. Figure 42 shows the Product Log interface.



Product Log

Product Log:

Category: printing

Product: Printer EPSON

Filter:

Filter By:

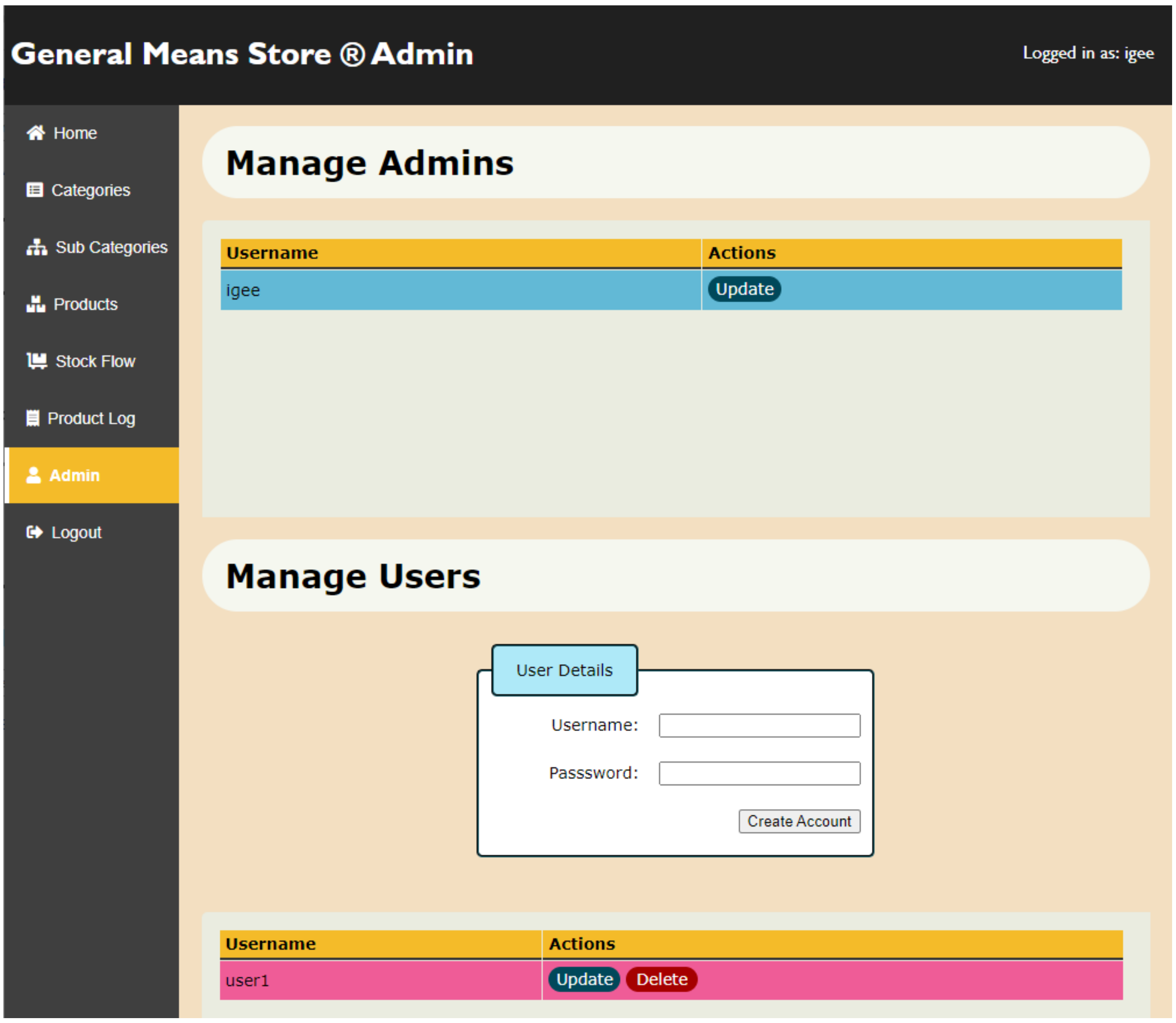
Year: Year

Destination: GMS

Product	Flow Type	Quantity	Supplier	Destination	Date	Time	In Stock	Inventory #
Bleach	exit	1		Power Department	2021-07-06	07:06:48	39	
Bleach	entry	40			2021-07-03	11:52:18	40	

Figure 45: Product Log Interface

Admin: In the admin page, the admin can manage admins and users accounts through updating or deleting them except the admin account which can be updated only. Besides, the admin can create user accounts as his desire. However, He can create no other admin accounts as the MMD head demand. The interface of this page is shown in Figure 43.



Manage Admins

Username	Actions
igee	Update

Manage Users

User Details

Username:

Passsword:

Username	Actions
user1	Update Delete

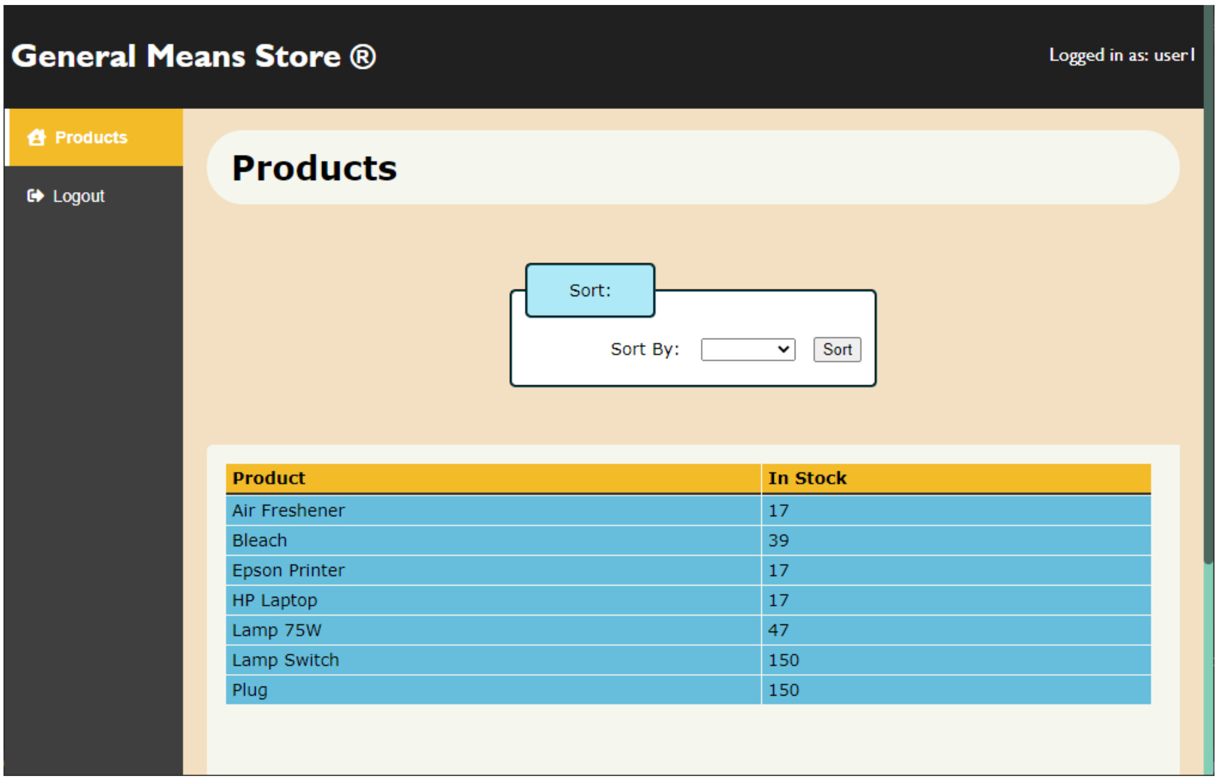
Figure 46 Admin Interface

Logout: The logout tool will end the admin session and logout from the application. After that the admin will need to login in order to access the application.

5.4.3 The Semipublic Area:

The semipublic area can be accessed by the users for whom the admin created an account. The tools in this area are described below.

Products: It shows the existing products and their corresponding stock to the user. Also it provide a sort tool to sort the content. Figure 44 shows its interface.



Product	In Stock
Air Freshener	17
Bleach	39
Epson Printer	17
HP Laptop	17
Lamp 75W	47
Lamp Switch	150
Plug	150

Figure 47: Products Interface

Logout: If the user clicks on the logout button then his session will be ended. He need to login again in case he want to access the application.

5.4.4 Mobile Interface:

In order to provide a responsive web design to the users of the application, I have configured the style for devices of screen width less than 480px using CSS media queries in a way, such that the application interface will be displayed properly when it is accessed from mobile browsers. Figure 45 shows the mobile Login page interface whereas Figure 46 show the Manage Products Mobile interface. Other interfaces are not shown since they are similar and have the same arrangement of the page content.

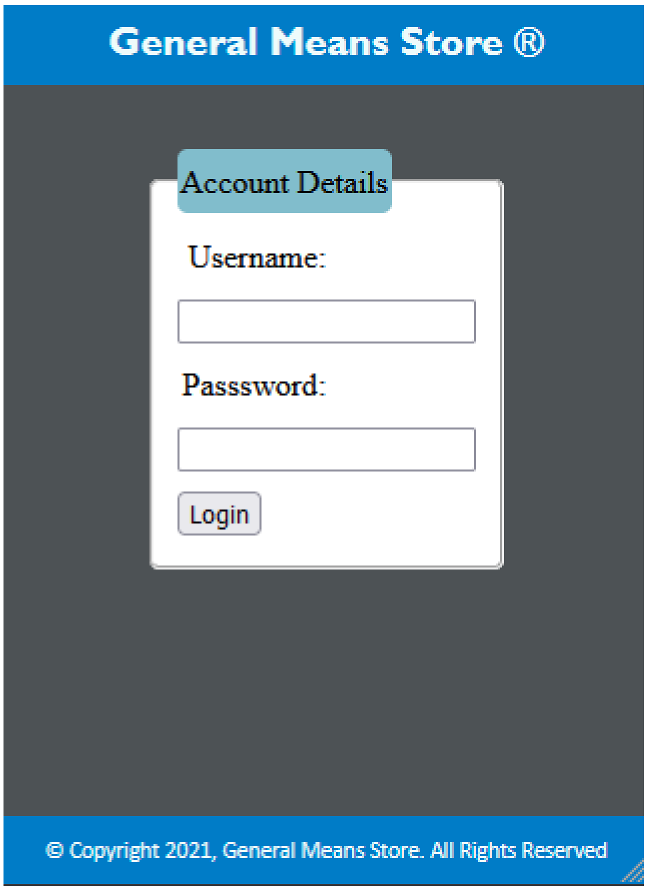
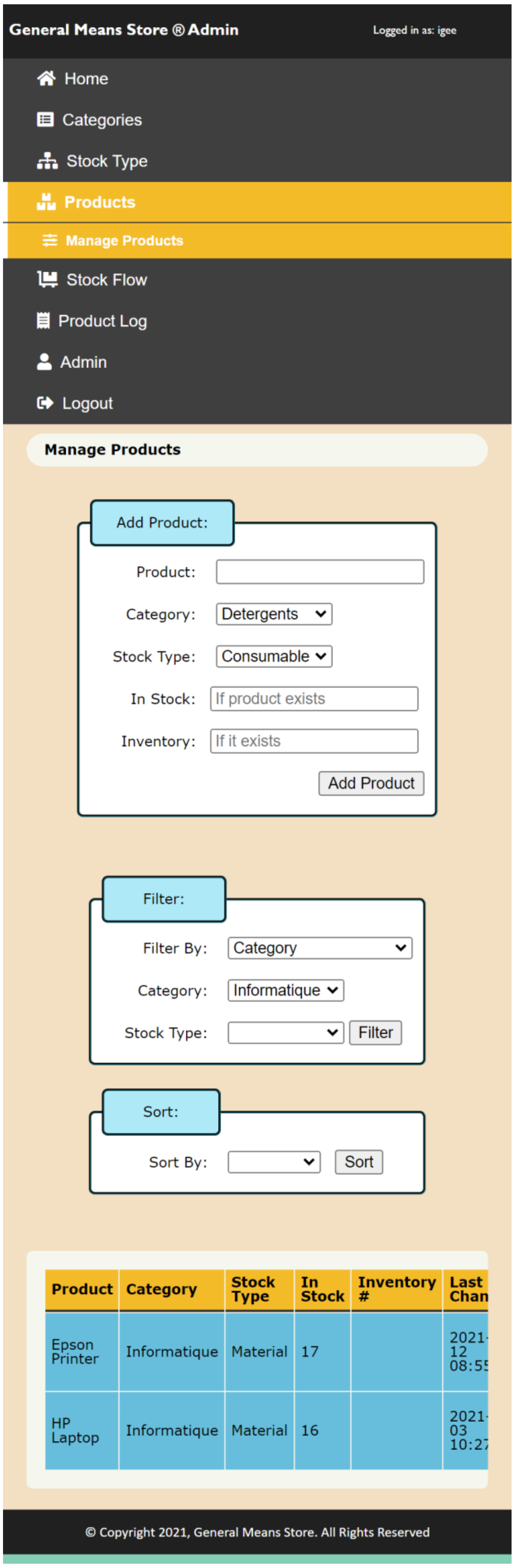


Figure 48: Mobile Login Interface.



General Means Store ® Admin

Logged in as: igee

Home

Categories

Stock Type

Products

Manage Products

Stock Flow

Product Log

Admin

Logout

Manage Products

Add Product:

Product:	
Category:	Detergents
Stock Type:	Consumable
In Stock:	If product exists
Inventory:	If it exists

Add Product

Filter:

Filter By:	Category
Category:	Informatique
Stock Type:	

Filter

Sort:

Sort By:

Sort

Product	Category	Stock Type	In Stock	Inventory #	Last Chan
Epson Printer	Informatique	Material	17		2021- 12 08:55
HP Laptop	Informatique	Material	16		2021- 03 10:27

© Copyright 2021, General Means Store. All Rights Reserved

Figure 49: Manage Products Mobile Interface

GENERAL CONCLUSION

In this report, I have presented the steps followed to design and develop the application intended for managing the GMS of our institute. I had analyzed the encountered problems in aim of providing a solutions to them. Moreover, I had gathered the requirements and information needed to achieve that purpose through several interviews with the staff members.

The built application solved the management problems since it allows the MD staff members, especially the storekeeper, to perform their tasks using the new technologies. Besides, it make the data stored electronically in the database, thus, data is secure and can be accessed easily and instantly. The implemented application can be accessed now using the internet from any location and device since it is a web application.

The experience of building this project was so useful and interesting. Its implementation leads me to gain new skills in the world of web development, and to become familiar with basic tools used in building websites.

FUTURE ENHANCEMENTS

Though the first version of this web application is implemented, it needs further work, enhancements and tools to be added to it. Thus, I recommend the following features as a future work:

- Enhancement of the interactivity of the application.
- The application should handle processing of orders.
- Provide a printing feature.
- More powerful security.
- Handling return process of defected products.
- Probabilistic and Statistical tools which will increase the efficiency of the management.

REFERENCES

[1] M. P. Dafydd Stuttard, The Web Application Hacker's Handbook: Discovering and Exploiting Security Flaws, Wiley Publishing, Inc, 2008.

[2] G. J. Ravi Das, Testing and Securing Web Applications, CRC Press, 2020.

[3] "Difference between Website and Web Application," [Online]. Available: https://www.guru99.com/difference-web-application-website.html. [Accessed 24 June 2021].

[4] C. Software, "The Benefits of Web-based Applications," 08 December 2015. [Online]. Available: https://coresolutions.ca/blogs/core-web/the-benefits-of-web-based-applications. [Accessed 25 June 2021].

[5] "Website: Static vs Dynamic - javatpoint," [Online]. Available: https://www.javatpoint.com/website-static-vs-dynamic. [Accessed 24 June 2021].

[6] J. N. Robbins, Learning Web Design a Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics, 5th ed., O'Reilly Media, Inc, 2018.

[7] D. Jon, HTML & CSS Design and Build Websites, John Wiley & Sons, Inc, 2011.

[8] A. S. Tanenbaum and D. J. Wetherall, Computer Networks, 5th ed., Pearson Education, Inc., publishing as Prentice Hall, 2011.

[9] J. F. Kurose and K. W. Ross, Computer Networking A Top-Down Approach, 7th ed., Pearson Education, Inc, 2017.

[10] R. Craig, "Why We Choose HTTPS over HTTP," 25 October 2018. [Online]. Available: https://coresolutions.ca/blogs/core-web/why-we-choose-https-over-http. [Accessed 25 June 2021].

[11] M. contributors, "What is a URL? - Learn web development," 01 June 2021. [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL. [Accessed 26 June 2021].

[12] Cloudflare, "What is a domain name? | Domain name vs. URL | Cloudflare," [Online]. Available: https://www.cloudflare.com/learning/dns/glossary/what-is-a-domain-name/. [Accessed 26 June 2021].

[13] M. Contrinutors, "IP Address - MDN Web Docs Glossary: Definitions of Web-related terms," 15 September 2020. [Online]. Available: https://developer.mozilla.org/en-US/docs/Glossary/IP_Address. [Accessed 26 June 2021].

[14] Kaspersky, "What is an IP Address & What does it mean?," [Online]. Available: https://www.kaspersky.com/resource-center/definitions/what-is-an-ip-address. [Accessed 26 June 2021].

[15] Cloudflare, "What is my IP address? | Cloudflare," [Online]. Available: https://www.cloudflare.com/learning/dns/glossary/what-is-my-ip-address/. [Accessed 26 June 2021].

[16] [Online]. Available: https://www.univ-eloued.dz/index.php/9-c-faculte-droit/2199-un-rapport-detaille-sur-les-moyens-d-interet-publi. [Accessed 28 June 2021].

[17] Wikipedia, "XAMPP - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/XAMPP. [Accessed 26 June 2021].

[18] Apache, "Welcome! - The Apache HTTP Server Project," [Online]. Available: https://httpd.apache.org/. [Accessed 27 June 2021].

[19] Guru99, "MariaDB vs MySQL: What is the Difference Between MariaDB and MySQL," [Online]. Available: https://www.guru99.com/mariadb-vs-mysql.html. [Accessed 27 June 2021].

[20] MariaDB, "About MariaDB Server - MariaDB.org," [Online]. Available: https://mariadb.org/about/. [Accessed 27 June 2021].

[21] phpMyAdmin.net, "phpMyAdmin," [Online]. Available: https://www.phpmyadmin.net/. [Accessed 27 June 2021].

[22] Wikipedia, "phpMyAdmin - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/PhpMyAdmin. [Accessed 27 June 2021].

[23] Makeuseof.com, "A Beginner's Guide to the Windows Command Prompt," [Online]. Available: https://www.makeuseof.com/tag/a-beginners-guide-to-the-windows-command-line/. [Accessed 27 June 2021].

[24] code.visualstudio, "Documentation for Visual Studio Code," [Online]. Available: https://code.visualstudio.com/docs. [Accessed 27 June 2021].

[25] code.visualstudio, "Get Started with Visual Studio Code," [Online]. Available: https://code.visualstudio.com/learn. [Accessed 27 June 2021].

[26] code.visualstudio, "IntelliSense in Visual Studio Code," [Online]. Available: https://code.visualstudio.com/docs/editor/intellisense. [Accessed 28 June 2021].

[27] Wikipedia, "Web browser - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Web_browser. [Accessed 28 June 2021].

[28] L. Lemay, R. Coburn and J. Kyrnin, SamsTeachYourself HTML, CSS & JavaScript Web Publishing, 7th ed., Pearson Education, Inc., 2016.

[29] mozilla.org, "What is a web browser?," [Online]. Available: https://www.mozilla.org/en-US/firefox/browsers/what-is-a-browser/. [Accessed 28 June 2021].

[30] F. Martinig, "StarUML - Open Source UML Tool," [Online]. Available: https://www.methodsandtools.com/tools/staruml.php. [Accessed 05 july 2021].

[31] "Introduction - StarUML documentation," [Online]. Available: https://docs.staruml.io/. [Accessed 05 july 2021].

[32] R. Blum, PHP, MySQL® & JavaScript® All-in-One For Dummies®, John Wiley & Sons, Inc., 2018.

[33] S. Suehring and J. Valade, PHP, MySQL®, JavaScript® & HTML5 All-in-One For Dummies®, John Wiley & Sons, Inc., 2013.

[34] R. Nixon, Learning PHP, MySQL & JavaScript, 5th ed., O'Reilly Media, Inc., 2018.

[35] J. Bush, Learn SQL Database Programming, Packt Publishing, 2020.

[36] R. Stephens, R. Plew and A. D. Jones, Sams Teach Yourself SQL in 24 Hours, 5th ed., Pearson Education, Inc., 2011.

[37] L. Rockoff, The Language of SQL, Course Technology, 2011.

[38] M. Knapp, HTML & CSS Learn The Fundamentals In 7 days, 2017.

[39] H. Gomaa, Software modeling and design : UML, use cases, patterns, and software architectures, Cambridge University Press, 2011.

[40] B. Selic, "The Theory and Practice of Modern Modeling Language Design for Model-based Software Engineering," in *Malina Software Corp*, 2011.

[41] J. Schmuller, Sams Teach Yourself UML in 24 Hours, 3rd ed., Sams Publishing, 2004.

[42] B. Unhelkar, Software Engineering with UML, CRC Press, Taylor & Francis Group, LLC, 2018.

[43] B. Oestereich, Developing software with UML, 2nd ed., Pearson Education Ltd, 2002.

[44] P. Roques, UML in Practice, John Wiley & Sons Ltd, 2004.

[45] R. Elmasri and S. B. Navathe, Fundamentals of Database Systems, 6th ed., Addison-Wesley (Pearson), 2011.