VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Santhibastawad Road, Machhe,
Belagavi - 590018, Karnataka, India

A
Mini project report on

"INVENTORY MANAGEMENT SYSTEM"

Submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

Submitted by

VISHAL K T (1JS20IS118)
VISHNU B V (1JS20IS119)

Under the guidance of

Dr. DAYANAND.P
Professor
Dept. of ISE, JSSATE
&
Mrs. NAGASHREE.S
Assistant Professor
Dept. of ISE, JSSATE

JSS ACADEMY OF TECHNICAL EDUCATION
JSSATEB Campus, Dr.Vishnuvardhan Road,
Uttarahalli Kengeri Main Road, Bengaluru-560060

2022-23

JSS ACADEMY OF TECHNICAL EDUCATION

JSSATEB Campus, Dr.Vishnuvardhan Road,
Uttarahalli Kengeri Main Road, Bengaluru-560060

DEPARTMENT OF INFORMATION SCIENCE & ENGINEERING

CERTIFICATE

Certified that the mini project work entitled **"INVENTORY MANAGEMENT SYSTEM"** carried out by **VISHAL K T(1JS20IS118) AND VISHNU B V(1JS20IS119)** are bona fide students of **JSS ACADEMY OF TECHICAL EDUCATION**, Bengaluru, in partial fulfillment for the award of degree of **BACHELOR OF ENGINEERING** of **VISVESVARAYA TECHNOLOGICAL UNIVERSITY,** Belagavi**,** during the academic year **2022-23**. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for the said degree.

Signature of Guide	**Signature of Guide**	**Signature of HOD**
Dr. DAYANAND.P	**Mrs. NAGASHREE.S**	**Dr. REKHA P M**
Professor	Assistant Professor	Professor & HOD
Dept. of ISE	Dept. of ISE	Dept. of ISE

External Viva

Name of the Examiners	**Signature and Date**
1.	
2.	

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible. So, with gratitude, I acknowledge all those whose guidance and encouragement crowned my effort with success.

First and foremost, I would like to express my sincere gratitude to his **Holiness Jagadguru Sri Sri Sri Shivarathri Deshikendra Mahaswamiji** for his divine blessings, without which the work wouldn’t have been possible.

It’s my pleasure in thanking our principal **Dr. Bhimasen Soragaon**, JSSATE, Bengaluru, for providing an opportunity to present this mini project as a part of our curriculum in the partial fulfillment of the degree.

I express my sincere gratitude to **Dr. Rekha P M**, Professor & Head of Department,Information Science and Engineering for her co-operation and encouragement at all moments of my approach.

It is my pleasant duty to place on record my deepest sense of gratitude to my guide **Dr. Dayanand P,** Professor, Dept. of Information Science and Engineering, and **Mrs. Nagashree S,** Asst. Professor, Dept. of Information Science and Engineering for the constant encouragement, valuable help, and assistance in every possible way.

VISHAL K T (1JS20IS118)
VISHNU B V (1JS20IS119)

ABSTRACT

Inventory management system is a project which aims in developing a computerized system to maintain the products availability and its related details. The information relates to the categories of products, buy price, selling price, product’s stock status and more. The system gives different access and different views to both user and admin. It has a facility where user can receive view the sales report of a timespan or update the sales tab.

The admin can view tabular data of highest selling products, latest sales and recently added products on his dashboard. Through user management he can manage the user’s role as admin or a normal user who is the employee of the company. He can also view the categories of products present and update the products. This project also makes it easy to add new products, delete a product and update number of products and other admin activities.

TABLE OF CONTENTS

1. INTRODUCTION

1.1 INTRODUCTION TO PROJECT

1.1.1 OVERVIEW

The "Inventory Management System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce thehardships faced by this existing system. More over this system is designed for the particular need of the company to carry out operations in a smooth and effective manner.

1.1.2 PROBLEM STATEMENT

To build a Inventory Management System that can efficiently handle all the functional activities of adding, removing and modifying the details of products.

1.1.3 SCOPE AND OBJECTIVES

The goal of this mini project is to develop a website for users to have a brief idea about the inventory status and the availability of products.

The main objectives of this website development can be defined asfollows:

1. To develop a system that provides functions to support users to sign in to their accounts.
2. To maintain records of product and its corresponding information such as buy price and selling price and more in a centralizeddatabase system.
3. To develop a system for user to be able to view the sales details.
4. To provide the users the functionalities of filtering out the required data and the ability to search for a specific data.

1.1.4 LIMITATIONS

The limitations of the periodic system include not knowing an exact inventory count in the middle of the period and running the risk of stockouts. With the periodic system, the company knows the inventory level with certainty only when it physically counts the inventory at the end of each period. Throughout the period, the company takes customer orders without knowing the exact inventory count or whether enough products are available to meet customer demand.
The other limitation is the values about the products can be mis entered causing the disruption in the flow of required products

1.2 INTRODUCTION TO DBMS

1.2.1 INTRODUCTION TO DATABASE AND DBMS

Database and database technology has a major impact on the growing use of computers. It is fair to say that databases play a critical role in almost all areas where computers are used, including business, electronic commerce, engineering, medicine, education, and library science. The word database is so commonly used that we must begin by defining what the database is.

Our initial definition is quite general. A database is a collection of related data. By data, we mean known facts that can be recorded and that have implicit meaning. For example, consider the names, telephone numbers, and addresses of the people you know. You may have recorded this data in an indexed address book, or you may have stored it on a hard drive, using personal computers and software such as Microsoft Excel. This collection of related data with an implicit meaning is a database.

The preceding definition of a database is quite general, for example, we may consider the collection of words that make up this page of text to be related data and hence to constitute a database. However, the common use of the term database is usually more restricted. A database has the following properties:

1. A database represents some aspect of the real world, sometimes called the mini world or the universe of discourse. The changes to the mini world are reflected in the database.
2. A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
3. A database is designed, built and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.

A database can be of any size and complexity. A database may be generated and maintained manually or computerized. For example, a library card catalog is a database that may be created and maintained manually. A database is a collection of data, typically describing the activities of one or more related organizations. For example, a university database might contain information about the following:

1. Entities such as students, faculty, courses, and classrooms.
2. Relationships between entities, such as student's enrollment in courses, faculty teaching courses, and the use of rooms for courses.

A database management system, or DBMS, is software designed to assist in maintaining and utilizing a large collection of data. The need for such systems as well as their use is growing rapidly.

DBMS provides an interface to perform various operations like database creation, storing data in it, updating data, creating a table in the database and a lot more. In the case of multiple users, it also maintains data consistency.

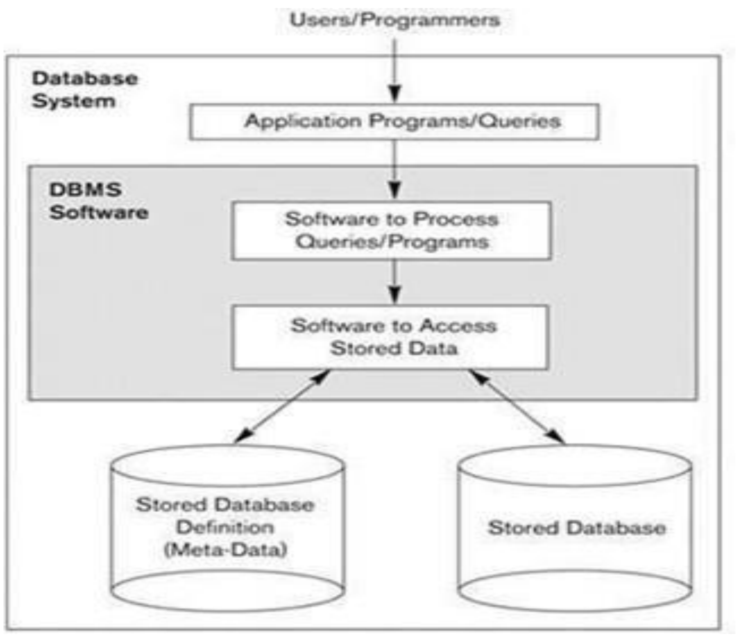

Users/Programmers
Database System
Application Programs/Queries
DBMS Software
Software to Process Queries/Programs
Software to Access Stored Data
Stored Database Definition (Meta-Data)
Stored Database

Figure 1: Simplified view of DBMS

1.2.2 ARCHITECTURE OF DBMS

Three schema architecture: The goal of the three-schema architecture illustrated in the figure is to separate the user application from the physical database. In this architecture, schemas can be defined at the following three levels:

1. The internal level has an internal schema, which describes the physical storage structure of the database. The internal schema uses a physical data model and describes thecomplete details of data storage and access paths for the database.
2. The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user

3. operations, and constraints. Usually, a representational data model is used to describe the conceptual schema when a database system is implemented. This implementation conceptual schema is often based on a conceptual schema design in a high-level data model.

4. The external or view level includes a number of external schemas or user views. Each external schema describes the part of a database that a particular user group is interested in and hides the rest of the database from that user group. As in the previous level, each external schema is typically implemented using a representational data model, possibly based on external schema design in a high-level data model.

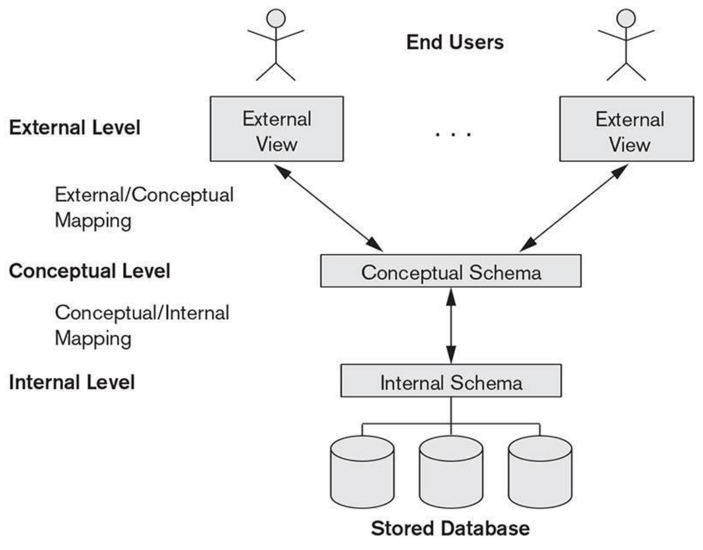


Figure 2: Three schema architecture of DBMS

1.2.3 ADVANTAGES OF USING DBMS

Using a DBMS to manage data has many advantages:

1. Data Independence: The application program should not, ideally, be expected to details of data representation and storage, the DBMS provides an abstract view of the data that hides such details.

2. Efficient Data Access: A DBMS utilizes a variety of sophisticated techniques to store and retrieve data efficiently. This feature is especially important if the data is to be stored on an external device.

3. Data Integrity and Security: If data is always accessed through DBMS, the DBMS can enforce integrity constraints. For example, before inserting salary information for an employee, the DBMS can check that the department budget is not exceeded. Also, it can enforce access controls that govern what data is visible to different classes of users.

4. Data Administration: When several users share data, centralizing the administration of data can offer significant improvements. Experienced professionals who understand the nature of the data being managed, and how different groups of users use it, it can be responsible for organizing the data representation to minimize redundancy and for fine- tuning the storage of the data to make retrieval efficient.

5. Concurrent Access and Crash Memory: A DBMS schedules concurrent access to the data in such a manner that users can think of the data as being accessed by only one user at a time. Further, the DBMS protects users from the effects of system failures.

6. Backup and Recovery: Database Management System automatically takes care of backup

and recovery. The users don't need to back up data periodically because this is taken care of by the DBMS. Moreover, it also restores the database after a crash or system failure to its previous condition.

7. Sharing of Data: In a database, the users of the database can share the data among themselves. There are various levels of authorization to access the data, and consequently the data can only be shared based on the correct authorization protocols being followed. Many remote users can also access the database simultaneously and share the databetween themselves.

2. REQUIREMENT SPECIFICATION

2.1 FUNCTIONAL REQUIREMENTS

The software is used for dog adoption management. The system should satisfy the following requirements:

1. Login option - The system must allow existing users to sign in to their accounts and continue using their account.
2. Add, remove, modify, the search for products - The system must allow the user to add,remove, modify and search for products .
3. Add, remove, modify, categories of products - The system must allow the user to Add, remove, modify, categories of products
4. View the sales report - The system must allow the user to View the sales report containing the products sold profit made and other important details for over a range of dates
5. Log out option - The system must allow existing users to Log out of their accounts and quit the application.

2.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are not mandatory, which means that they are not compulsory to be fulfilled. Some non-functional requirements of the system are:

1. The system must be portable and should work on most of the platforms.
2. Each page must load within 2 seconds.
3. The system must be display accurate data without any errors.
4. Database security must meet the standard requirements.

2.3 HARDWARE REQUIREMENTS

1. A desktop or laptop with all necessary hardware components like mouse, keyboard and a stable internet connection.
2. 50GB+ of the hard disk.
3. 4GB+ of the RAM.
4. Any browser capable performing basic required operation.

2.4 SOFTWARE REQUIREMENTS

1. Operating System: Windows Vista/7/8/10
2. Front End: Html, CSS, Javascript
3. Backend: PHP
4. Database: SQL
5. Web Server: XAMPP SERVER

2.4.1 HTML

Hypertext Mark-up Language (**HTML**) is the standard mark-up for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img /> and <input /> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may include other tags as sub-elements.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

2.4.2 JAVA SCRIPT

JavaScript, abbreviated as JS, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it for client-side page behavior, and all major web browsers have a dedicated JavaScript engine to execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). JavaScript engines were originally used only in web browsers, but they are now embedded in some servers, usually via Node.js. They are also embedded in a variety of applications created with frameworks such as Electron and Cordova.

2.4.3 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a mark-up language like HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content also makes it feasible to present the same mark-up page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name *cascading* comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

2.4.4 MySQL

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. It is developed, marketed and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons −It is released under an open-source license. So, you have nothing to pay to use it, it is a very powerful program in its own right and handles a large subset of the functionality of the most expensive and powerful database packages. MySQL uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc and works very quickly and works well even with large data sets.

2.4.5 PHP

PHP Hypertext Preprocessor is a scripting language that helps people make web pages more interactive by allowing them to do more things. PHP code is run on the web server. A website programmed with PHP can have pages that are password protected. A website with no programming cannot do this without other complex things. Standard PHP file extensions are: .php .php3 or .phtml, but a web server can be set up to use any extension. Its structure was influenced by many languages like C, Perl, Java, C++, and even Python. It is considered to be free software by the Free Software Foundation.

3. SYSTEM DESIGN

3.1 DATABASE DESIGNER

System is a collection of interrelated components that work together to achieve a purpose. System analysis refers to the systematic examination or detailed study of a system in order to identify problems of the system, and using the information gathered in the analysis stage to recommend improvements or solutions to the system.

System design is an abstract representation of a system component and their relationship and which describe the aggregated functionality and performance of the system. System design is also the plan or blueprint for how to obtain an answer to the question being asked. The design specifies which of the various types of approach. Systems design is the process or art of defining the architecture, components modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development.

Database Design is a collection of processes that facilitate the designing, development, implementation and maintenance of enterprise data management systems. Properly designed databases are easy to maintain, improve data consistency and are cost effective in terms of disk storage space. The database designer decides how the data elements correlate and what data must be stored. The main objectives of database designing are to produce logical and physical designs models of the proposed database system.

ATTRIBUTES

Attributes define the properties of a data object and take on one of three different characteristics. They can be used to:

- Name an instance of a data object.
- Describe the instance.

The following figure 3 .1 shows the Relational model for Inventory Management System.

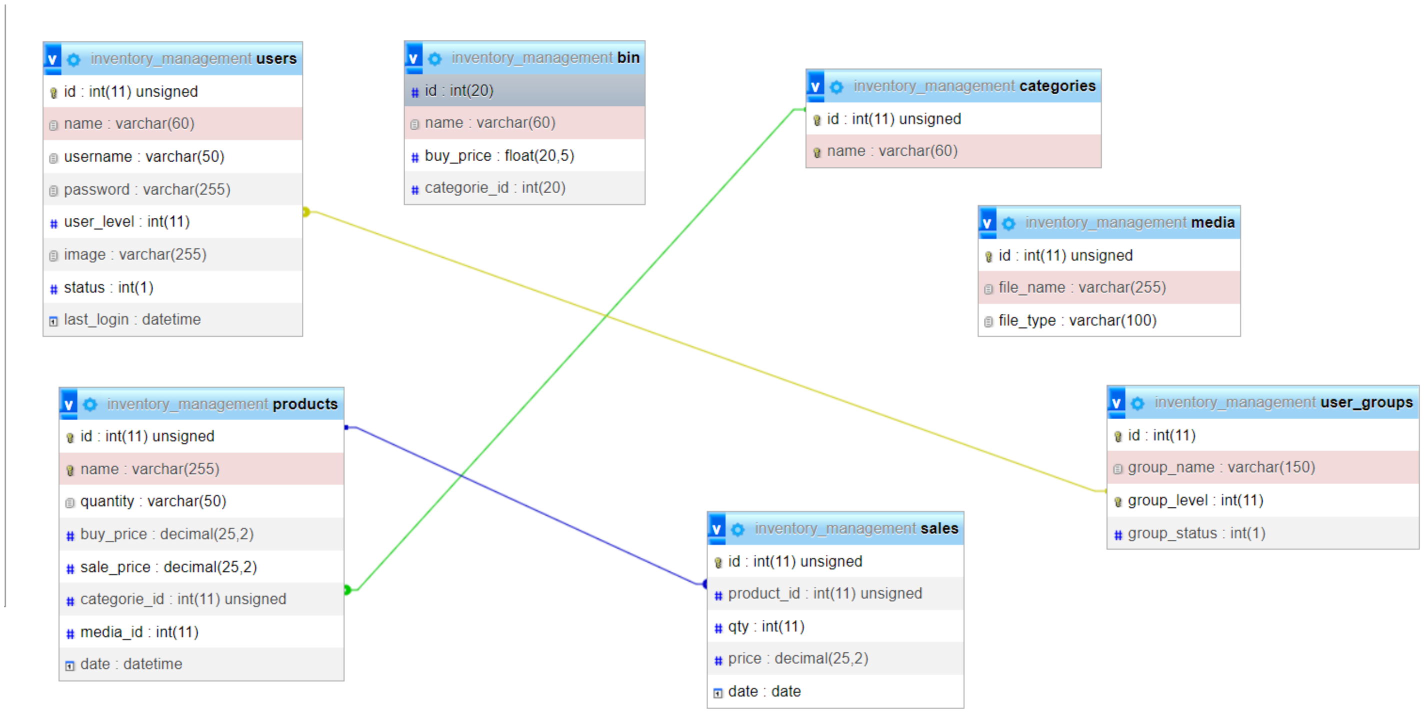


Figure 3.1: Database designer

The Relational Model was proposed by E.F. Codd to model data in the form of relations or tables. After designing the conceptual model of Database using an ER diagram, we need to convert the conceptual model into the relational model which can be implemented using any RDBMS languages like Oracle SQL, MySQL etc. So, we will see what the Relational Model is. Relational Model represents how data is stored in Relational Databases. A relational database stores data in the form of relations (tables).

3.2 ENTITY RELATIONSHIP DIAGRAM

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

It allows us to describe the data involved in a real-world enterprise in terms of objects and their relationship, widely used to develop an initial database design. By defining the entities, their attributes, and showing the relationships between them, an ER diagram illustrates the logical structure of databases. ER diagrams are used to sketch out the design of a database.

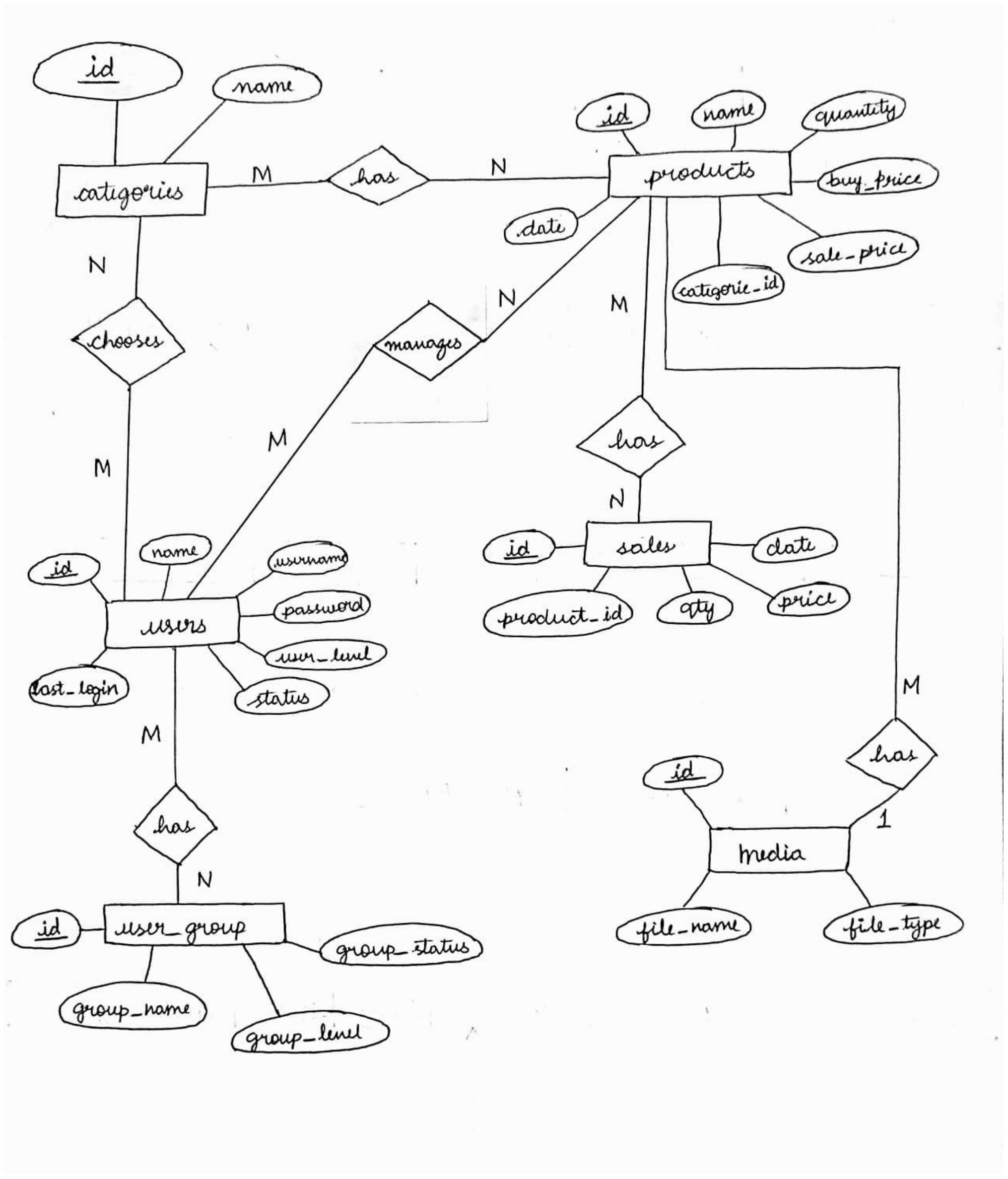


Figure 3.2: Entity relation diagram

Rectangle: Represents Entity sets

Ellipses: Attributes

Diamonds: Relationship Set

Lines: They link attributes to Entity Sets and Entity sets to Relationship Set

Double Ellipses: Multivalued Attributes

Dashed Ellipses: Derived Attributes

Double Rectangles: Weak Entity Sets

Double Lines: Total participation of an entity in a relationship set

Relations

M: N

- There can be multiple user groups and multiple users in each group
- There can be multiple product categories and multiple products in each category
- There can be multiple users and each user can manage multiple products
- There can be multiple users and each user can choose multiple product categories
- There can be multiple products and each product can have multiple sales

M:1

- There can be multiple products but each product can have only 1 media file associated with it

3.3 SCHEMA DIAGRAM

A **database schema** is the skeleton structure that represents the logical view of the entire database. A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams.

A **schema diagram** contains entities and the attributes that will define that **schema**. It only shows us the database design. It does not show the actual data of the database. **Schema** can be a single table or it can have more than one table which is related.

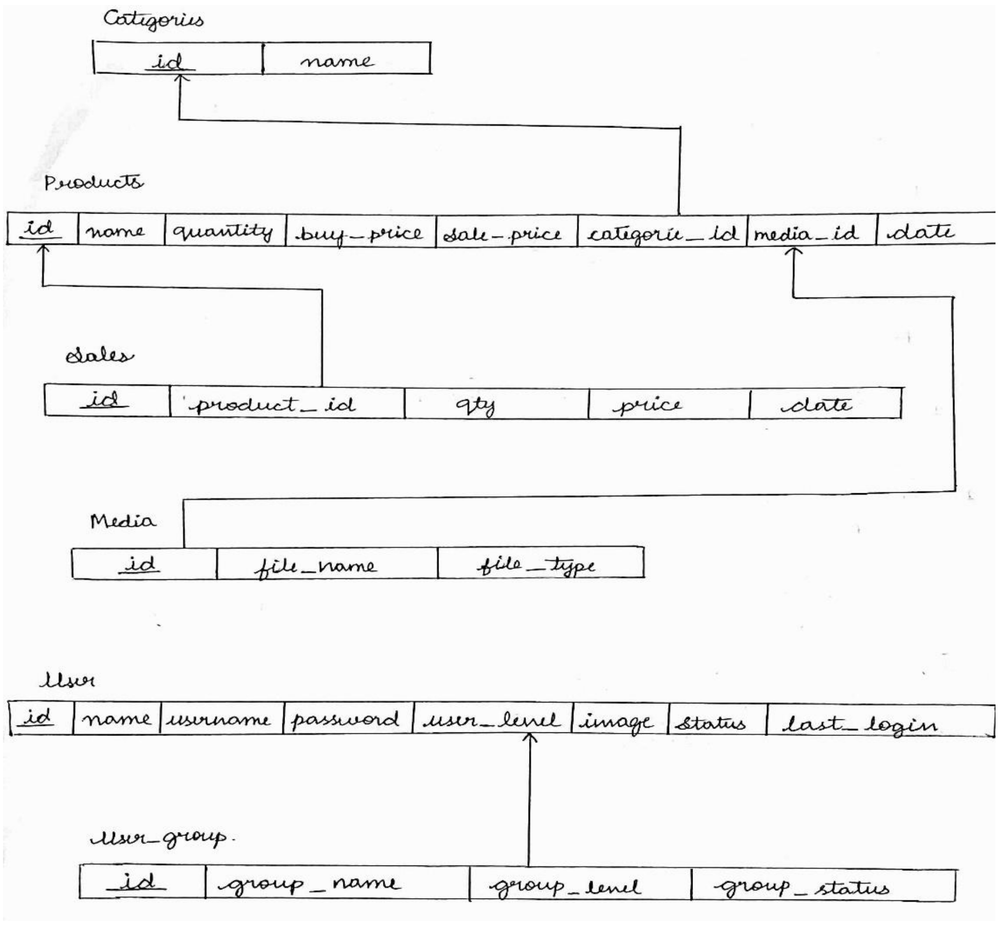


Figure 3.3: Schema diagram

The schema diagram above represents different tables used, and underlined attributes are primary keys and arrows are used to represent foreign keys.

3.4 TABLES

3.4.1 CATEGORIES

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	**Primary key, unsigned not null**	**int (11)**
name	**Not null**	**Varchar(60)**

3.4.2 Products

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	**Primary key, unsigned not null**	**int (11)**
name	**Not null**	**Varchar(60)**
Quantity	**Default null**	**Varchar(50)**
Buy_price	**Default null**	**Decimal(25,2)**
Sale_price	**Default null**	**Decimal(25,2)**
Categorie_id	**Unsigned not null**	**Int(11)**
Media_id	**Default 0**	**Int(11)**
date	**Not null**	**Datetime**

3.4.3 SALES

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	Primary key, unsigned not null	int (11)
Product_id	Unsigned not null	Int(11)
Qty	Not null	Decimal(25,2)
Price	Not null	Decimal(25,2)
date	Not null	date

3.4.4 USERS

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	Primary key, unsigned not null	int (11)
name	Not null	Varchar(60)
Username	Not null	Varchar(60)
password	Not null	Varchar(60)
User_level	Not null	Int(11)
Image	Default ‘no_image.jpg’	Varchar(255)
Status	Not null	Int(1)
Last_Login	Default null	datetime

3.4.5 MEDIA

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	Primary key, unsigned not null	int (11)
File_name	Not null	Varchar(255)
File_type	Not null	Varchar(100)

3.4.6 USER GROUPS

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	Primary key, unsigned not null	int (11)
Group_name	Not null	Varchar (255)
Group_level	Not null	int (11)
Group_status	Not null	int (1)

3.4.7 BIN

ATTRIBUTES	CONSTRAINTS	DATA TYPE
id	Primary key, unsigned not null	int (11)
name	Not null	Varchar (60)
Buy_price	Default null	Decimal (25,2)
Categorie_id	Unsigned not null	Int (11)

3.5 NORMALIZATION

Database normalization is a database schema design technique, by which an existing schema is modified to minimize redundancy and dependency of data. Normalization splits a large table into smaller tables and defines relationships between them to increase the clarity in organizing data.
Database normalization types

First Normal Form (1NF)

• First normal form (1NF) deals with the `shape' of the record type

• A relation is in 1NF if, and only if, it contains no repeating attributes or groups of attributes.

• Example: The Student table with the repeating group is not in 1NF

Second Normal Form (2NF)

• A relation is in 2NF if, and only if, it is in 1NF and every non-key attribute is fully functionally dependent on the whole key.

Third Normal Form (3NF)

• A relation is in 3NF if, and only if, it is in 2NF and there are no transitive functional dependencies.

• Transitive functional dependencies arise:

• when one non-key attribute is functionally dependent on another non-key attribute

• FD: non-key attribute -> non-key attribute

• and when there is redundancy in the database

Boyce-Codd Normal Form (BCNF)

• When a relation has more than one candidate key, anomalies may result even though the relation is in 3NF.

• 3NF does not deal satisfactorily with the case of a relation with overlapping candidate keys

• i.e., composite candidate keys with at least one attribute in common.

• BCNF is based on the concept of a determinant.

Fourth Normal Form (4NF)

• It is a normal form used in database normalization Introduced by Ronald Fagin in 1977, 4NF is the next level of normalization after Boyce–Codd normal form (BCNF). Whereas the second, third, and Boyce–Codd normal forms are concerned with functional dependencies, 4NF is concerned with a more general type of dependency known as a multivalued dependency

Fifth Normal Form (5NF)

• It is also known as project-join normal form (PJ/NF) is a level of database normalization designed to reduce redundancy in relational databases recording multi-valued facts by isolating semantically related multiple relationships. A table is said to be in the 5NF if and only if every non trivial join dependency in that table is implied by the candidate keys.

3.6 QUERIES

The below mentioned are all the queries used to perform various tasks in MySQL

Query to find all database table rows by table name

```
--------------------------------------------------------------------------
function find_all($table) {
   global $db;
   if(tableExists($table))
   {
     return find_by_sql("SELECT * FROM ".$db->escape($table));
   }
}
```

Query to Find data from table by id

```
----------------------------------------------------
function find_by_id($table,$id)
{
  global $db;
  $id = (int)$id;
    if(tableExists($table)){
   $sql = $db->query("SELECT * FROM {$db->escape($table)} WHERE id='{$db-
>escape($id)}' LIMIT 1");
          if($result = $db->fetch_assoc($sql))
            return $result;
          else
            return null;
     }
}
```

Query to Delete data from table by id

```
----------------------------------------------------
function delete_by_id($table,$id)
{
  global $db;
  if(tableExists($table))
   {
    $sql = "DELETE FROM ".$db->escape($table);
    $sql .= " WHERE id=". $db->escape($id);
    $sql .= " LIMIT 1";
    $db->query($sql);
return ($db->affected_rows() === 1) ? true : false;
   }
}
```

Query to Count id By table name

```
-----------------------------------------------------
function count_by_id($table){
  global $db;
  if(tableExists($table))
  {
  $sql    = "SELECT COUNT(id) AS total FROM ".$db->escape($table);
    $result = $db->query($sql);
     return($db->fetch_assoc($result));
  }
}
```

Query to Determine if database table exists

```
-----------------------------------------------------
function tableExists($table){
  global $db;
  $table_exit = $db->query('SHOW TABLES FROM '.DB_NAME.' LIKE "'.$db-
>escape($table).'"');
      if($table_exit) {
        if($db->num_rows($table_exit) > 0)
              return true;
         else
              return false;
      }
  }
```

Query to Login with the data provided in $_POST, coming from the login form.

```
-----------------------------------------------------
  function authenticate($username='', $password='') {
    global $db;
```

```
 $username = $db->escape($username);
    $password = $db->escape($password);
    $sql  = sprintf("SELECT id,username,password,user_level FROM users WHERE username ='%s'
LIMIT 1", $username);
    $result = $db->query($sql);
    if($db->num_rows($result)){
      $user = $db->fetch_assoc($result);
      $password_request = sha1($password);
      if($password_request === $user['password'] ){
        return $user['id'];
      }
    }
   return false;
  }
```

Query to Login with the data provided in $_POST, coming from the login_v2.php form. If you used this method then remove authenticate function.

```
 ----------------------------------------------------
   function authenticate_v2($username='', $password='') {
     global $db;
     $username = $db->escape($username);
     $password = $db->escape($password);
     $sql  = sprintf("SELECT id,username,password,user_level FROM users WHERE username
     ='%s' LIMIT 1", $username);
     $result = $db->query($sql);
     if($db->num_rows($result)){
       $user = $db->fetch_assoc($result);
       $password_request = sha1($password);
       if($password_request === $user['password'] ){
         return $user;
       }
```

```
}
     return false;
    }
```

Query to Find current log in user by session id

```
  function current_user(){
      static $current_user;
      global $db;
      if(!$current_user){

     if(isset($_SESSION['user_id'])):
           $user_id = intval($_SESSION['user_id']);
           $current_user = find_by_id('users',$user_id);
        endif;
      }
    return $current_user;
  }
```

Query to update the last log in of a user

```
 function updateLastLogIn($user_id)
     {
             global $db;
             $date = make_date();
               $sql = "UPDATE users SET last_login='{$date}' WHERE id ='{$user_id}' LIMIT 1";
               $result = $db->query($sql);

return ($result && $db->affected_rows() === 1 ? true : false);
     }
```

Query to Find all Group name

```
------------------------------------------------------
  function find_by_groupName($val)
  {
    global $db;
    $sql = "SELECT group_name FROM user_groups WHERE group_name = '{$db->escape($val)}'
LIMIT 1 ";
    $result = $db->query($sql);
    return($db->num_rows($result) === 0 ? true : false);
  }
```

Query to Find group level

```
------------------------------------------------------
  function find_by_groupLevel($level)
  {
    global $db;
    $sql = "SELECT group_level FROM user_groups WHERE group_level = '{$db-
>escape($level)}' LIMIT 1 ";
    $result = $db->query($sql);
    return($db->num_rows($result) === 0 ? true : false);
  }
```

Query to checking which user level has access to page

```
  ------------------------------------------------------
   function page_require_level($require_level){
     global $session;
  $current_user = current_user();
     $login_level = find_by_groupLevel($current_user['user_level']);
     //if user not login
     if (!$session->isUserLoggedIn(true)):
```

```
 $session->msg('d','Please login...');
          redirect('index.php', false);
     //if Group status Deactive
    elseif($login_level['group_status'] === '0'):
          $session->msg('d','This level user has been band!');
          redirect('home.php',false);
     //cheackin log in User level and Require level is Less than or equal to
    elseif($current_user['user_level'] <= (int)$require_level):
             return true;
     else:
          $session->msg("d", "Sorry! you dont have permission to view the page.");
          redirect('home.php', false);
       endif;
  }
```

Query to Finding all product name , Request coming from ajax.php for auto suggest

----

```
   function find_product_by_title($product_name){
    global $db;
    $p_name = remove_junk($db->escape($product_name));
    $sql = "SELECT name FROM products WHERE name like '%$p_name%' LIMIT 5";
    $result = find_by_sql($sql);
    return $result;
  }
```

Query to Finding all product info by product title Request coming from ajax.php

```
--------------------------------------------------
 function find_all_product_info_by_title($title){
   global $db;
   $sql  = "SELECT * FROM products ";
   $sql .= " WHERE name ='{$title}'";
   $sql .=" LIMIT 1";
   return find_by_sql($sql);
 }
```

Query to Update product quantity

```
--------------------------------------------------
 function update_product_qty($qty,$p_id){
   global $db;
   $qty = (int) $qty;
   $id  = (int)$p_id;
   $sql = "UPDATE products SET quantity=quantity -'{$qty}' WHERE id = '{$id}'";
   $result = $db->query($sql);
   return($db->affected_rows() === 1 ? true : false);
 }
```

Query to Generate sales report by two dates

```
--------------------------------------------------
function find_sale_by_dates($start_date,$end_date){
 global $db;
 $start_date  = date("Y-m-d", strtotime($start_date));
 $end_date    = date("Y-m-d", strtotime($end_date));
    $sql  = "SELECT s.date, p.name,p.sale_price,p.buy_price,";
 $sql .= "COUNT(s.product_id) AS total_records,";
```

```
  $sql .= "SUM(s.qty) AS total_sales,";
  $sql .= "SUM(p.sale_price * s.qty) AS total_saleing_price,";
  $sql .= "SUM(p.buy_price * s.qty) AS total_buying_price ";
  $sql .= "FROM sales s ";
  $sql .= "LEFT JOIN products p ON s.product_id = p.id";
  $sql .= " WHERE s.date BETWEEN '{$start_date}' AND '{$end_date}'";
  $sql .= " GROUP BY DATE(s.date),p.name";
  $sql .= " ORDER BY DATE(s.date) DESC";
  return $db->query($sql);
}
```

Query to Generate Daily sales report

```
----------------------------------------------------
function  dailySales($year,$month){
  global $db;
  $sql  = "SELECT s.qty,";
  $sql .= " DATE_FORMAT(s.date, '%Y-%m-%e') AS date,p.name,";
  $sql .= "SUM(p.sale_price * s.qty) AS total_saleing_price";
  $sql .= " FROM sales s";
  $sql .= " LEFT JOIN products p ON s.product_id = p.id";
  $sql .= " WHERE DATE_FORMAT(s.date, '%Y-%m' ) = '{$year}-{$month}'";
  $sql .= " GROUP BY DATE_FORMAT( s.date,  '%e' ),s.product_id";
  return find_by_sql($sql);
}
```

Query to Generate Monthly sales report

```
----------------------------------------------------
function  monthlySales($year){
  global $db;
  $sql  = "SELECT s.qty,";
 $sql .= " DATE_FORMAT(s.date, '%Y-%m-%e') AS date,p.name,";
  $sql .= "SUM(p.sale_price * s.qty) AS total_saleing_price";
 $sql .= " FROM sales s";
  $sql .= " LEFT JOIN products p ON s.product_id = p.id";
  $sql .= " WHERE DATE_FORMAT(s.date, '%Y' ) = '{$year}'";
  $sql .= " GROUP BY DATE_FORMAT( s.date,  '%c' ),s.product_id";
  $sql .= " ORDER BY date_format(s.date, '%c' ) ASC";
  return find_by_sql($sql);
}
```

EDITING THE TABLES

Query to ADD GROUPS

```
 $query  = "INSERT INTO user_groups (";
        $query .="group_name,group_level,group_status";
        $query .=") VALUES (";
        $query .=" '{$name}', '{$level}','{$status}'";
        $query .=")";
```

Query to ADD PRODUCT

```
$query  = "INSERT INTO products (";
     $query .=" name,quantity,buy_price,sale_price,categorie_id,media_id,date";
     $query .=") VALUES (";
     $query .=" '{$p_name}', '{$p_qty}', '{$p_buy}', '{$p_sale}', '{$p_cat}', '{$media_id}', '{$date}'";
$query .=")";
 $query .=" ON DUPLICATE KEY UPDATE name='{$p_name}'";
```

Query to ADD SALES

```
$sql  = "INSERT INTO sales (";
          $sql .= " product_id,qty,price,date";

 $sql .= ") VALUES (";
          $sql .= "'{$p_id}','{$s_qty}','{$s_total}','{$s_date}'";
          $sql .= ")";
```

Query to ADD USER

```
$query = "INSERT INTO users (";
        $query .="name,username,password,user_level,status";
        $query .=") VALUES (";
        $query .=" '{$name}', '{$username}', '{$password}', '{$user_level}','1'";
        $query .=")";
```

Query to INSERT TO CATAGORY

```
$sql  = "INSERT INTO categories (name)";
      $sql .= " VALUES ('{$cat_name}')";
```

Query to UPDATE PASSWORD

```
 $sql = "UPDATE users SET password ='{$new}' WHERE id='{$db->escape($id)}'";
```

Query to EDIT ACCOUNT

```
$sql = "UPDATE users SET name ='{$name}', username ='{$username}' WHERE id='{$id}'";
```

Query to EDIT CATAGORY

```
 $sql = "UPDATE categories SET name='{$cat_name}'";
       $sql .= " WHERE id='{$categorie['id']}'";
```

Query to EDIT GROUP

```
$query  = "UPDATE user_groups SET ";
     $query .= "group_name='{$name}',group_level='{$level}',group_status='{$status}'";
        $query .= "WHERE ID='{$db->escape($e_group['id'])}'";
```

Query to EDIT PRODUCT

```
$query   = "UPDATE products SET";
       $query  .=" name ='{$p_name}', quantity ='{$p_qty}',";
       $query  .=" buy_price ='{$p_buy}', sale_price ='{$p_sale}', categorie_id
='{$p_cat}',media_id='{$media_id}'";
       $query  .=" WHERE id ='{$product['id']}'";
```

Query to EDIT SALES

```
$sql  = "UPDATE sales SET";
          $sql .= " product_id= '{$p_id}',qty={$s_qty},price='{$s_total}',date='{$s_date}'";
          $sql .= " WHERE id ='{$sale['id']}'";
```

Query to EDIT USER

```
$sql = "UPDATE users SET name ='{$name}', username
='{$username}',user_level='{$level}',status='{$status}' WHERE id='{$db->escape($id)}'";
```

JOINS

Query to Find all user by Joining users table and user groups table

```
----------------------------------------------------
 function find_all_user(){
     global $db;
     $results = array();
     $sql = "SELECT u.id,u.name,u.username,u.user_level,u.status,u.last_login,";
     $sql .="g.group_name ";
     $sql .="FROM users u ";
     $sql .="LEFT JOIN user_groups g ";
     $sql .="ON g.group_level=u.user_level ORDER BY u.name ASC";
     $result = find_by_sql($sql);
     return $result;
 }
```

Query to Finding all product name JOIN with categorie and media database table

```
  ----------------------------------------------------
 function join_product_table(){
    global $db;
    $sql  =" SELECT p.id,p.name,p.quantity,p.buy_price,p.sale_price,p.media_id,p.date,c.name";
   $sql  .=" AS categorie,m.file_name AS image";
  $sql  .=" FROM products p";
   $sql  .=" LEFT JOIN categories c ON c.id = p.categorie_id";
   $sql  .=" LEFT JOIN media m ON m.id = p.media_id";
   $sql  .=" ORDER BY p.id ASC";
   return find_by_sql($sql);
```

Query to Display Recent product Added

```
---------------------------------------------------
function find_recent_product_added($limit){
  global $db;
  $sql   = " SELECT p.id,p.name,p.sale_price,p.media_id,c.name AS categorie,";
  $sql  .= "m.file_name AS image FROM products p";
  $sql  .= " LEFT JOIN categories c ON c.id = p.categorie_id";
  $sql  .= " LEFT JOIN media m ON m.id = p.media_id";
  $sql  .= " ORDER BY p.id DESC LIMIT ".$db->escape((int)$limit);
  return find_by_sql($sql);
 }
```

Query to Find Highest selling Product

```
---------------------------------------------------
function find_higest_saleing_product($limit){
  global $db;
  $sql  = "SELECT p.name, COUNT(s.product_id) AS totalSold, SUM(s.qty) AS totalQty";
  $sql .= " FROM sales s";
  $sql .= " LEFT JOIN products p ON p.id = s.product_id ";
  $sql .= " GROUP BY s.product_id";
  $sql .= " ORDER BY SUM(s.qty) DESC LIMIT ".$db->escape((int)$limit);
  return $db->query($sql);
 }
```

Query To find all sales

```
---------------------------------------------------
function find_all_sale(){
  global $db;
  $sql  = "SELECT s.id,s.qty,s.price,s.date,p.name";
  $sql .= " FROM sales s";
```

```
 $sql .= " LEFT JOIN products p ON s.product_id = p.id";
   $sql .= " ORDER BY s.date DESC";
   return find_by_sql($sql);
 }
```

Query To Display Recent sale

```
function find_recent_sale_added($limit){
  global $db;
  $sql  = "SELECT s.id,s.qty,s.price,s.date,p.name";
  $sql .= " FROM sales s";
  $sql .= " LEFT JOIN products p ON s.product_id = p.id";
  $sql .= " ORDER BY s.date DESC LIMIT ".$db->escape((int)$limit);
  return find_by_sql($sql);
}
```

3.7 TRIGGERS

A trigger is a special type of stored procedure that automatically runs when an event occurs in the database server. DML triggers run when a user tries to modify data through a data manipulation language (DML) event. DML events are INSERT, UPDATE, or DELETE statements on a table or view.

Syntax: create trigger [trigger_name] [before | after]
{insert | update | delete}
on [table_name]
[for each row] [trigger_body]

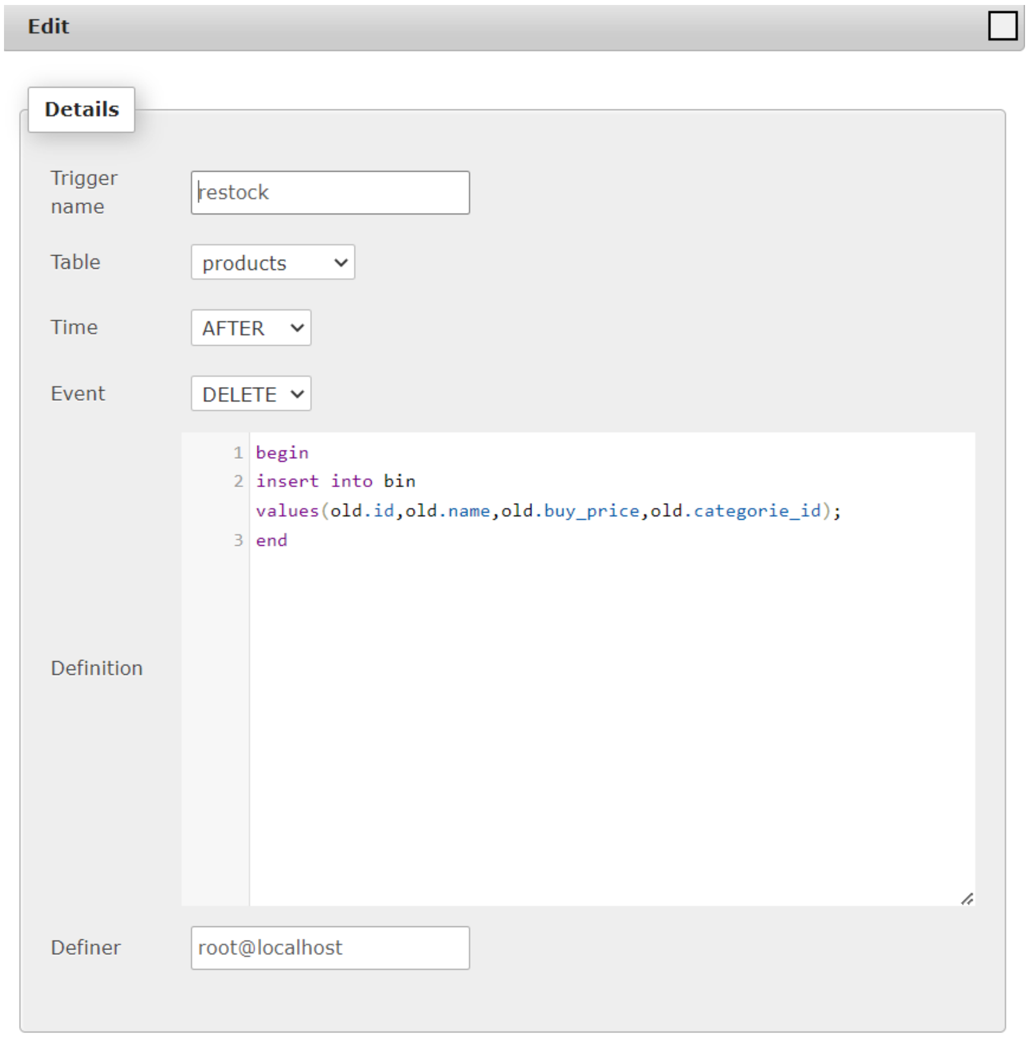
Trigger For Backing Up

Code:

```
begin
insert into bin values(old.id,old.name,old.buy_price,old.categorie_id);
end
```

Description:

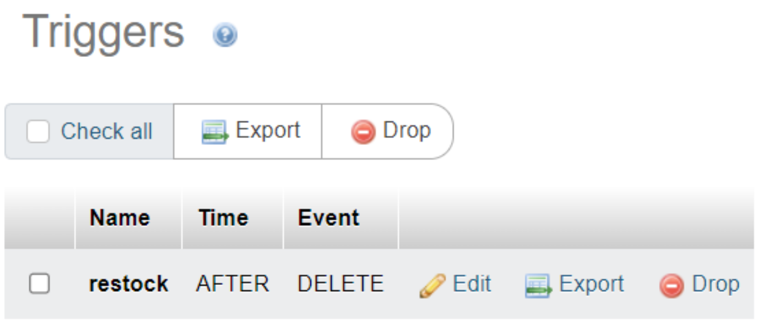
In this trigger we aim to back up the product on deleting it from the product table
The product is deleted from the products table when its quantity reaches zero.
When this is done the product on deleting from the product tables is added into the bin table which acts a backup table.
This table can be viewwd later to see the products which are out of stock and can be restocked and removed from the bin and added back to the products table



Trigger name	restock
Table	products
Time	AFTER
Event	DELETE
Definition	
Definer	root@localhost

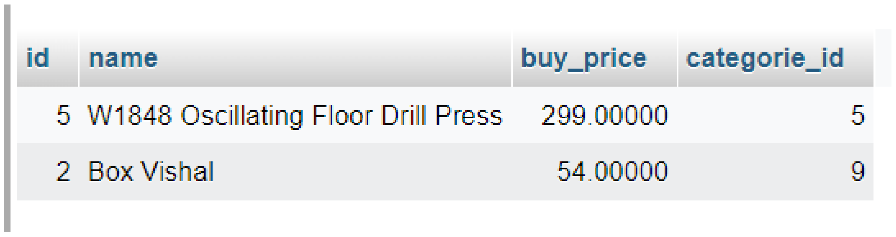
```
begin
insert into bin
values(old.id,old.name,old.buy_price,old.categorie_id);
end
```

Fig 3.7.1 Shows the implementation of trigger which is executed when product is deleted



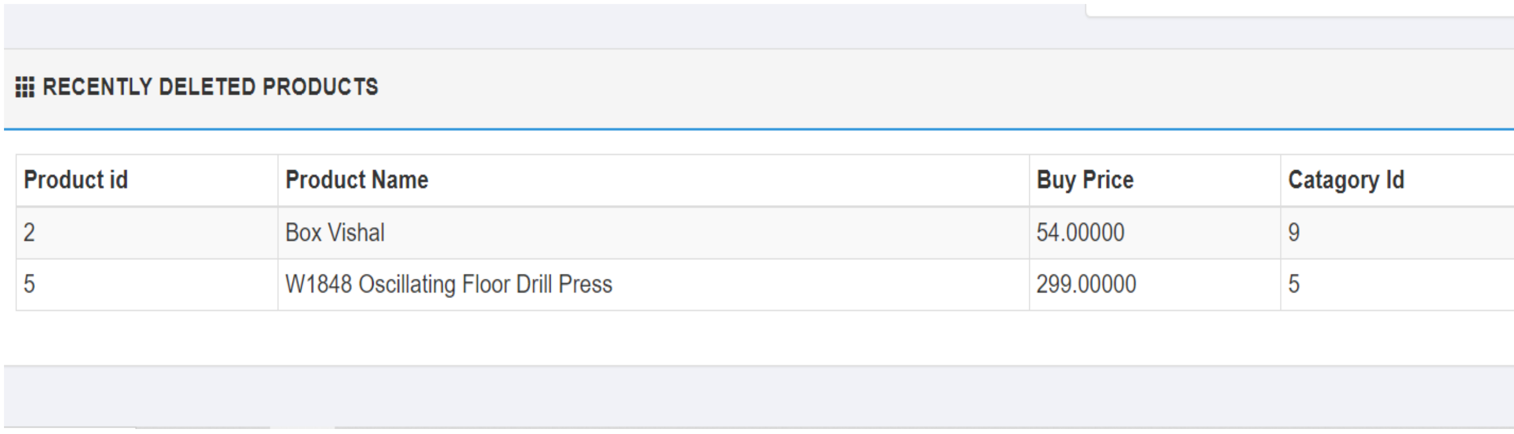
	Name	Time	Event	
☐	**restock**	AFTER	DELETE	Edit Export Drop

Fig 3.7.2 Shows the trigger present in the products table



id	name	buy_price	categorie_id
5	W1848 Oscillating Floor Drill Press	299.00000	5
2	Box Vishal	54.00000	9

Fig 3.7.3 Shows the bin table which gets updated on deletion of product from products table



RECENTLY DELETED PRODUCTS

Product id	Product Name	Buy Price	Catagory Id
2	Box Vishal	54.00000	9
5	W1848 Oscillating Floor Drill Press	299.00000	5

Fig 3.7.4 Shows the working of trigger in the front end

4 IMPLEMENTATION AND EXPERIMENTAL RESULTS

Implementation is the stage where the theoretical design is turned into a working system. The most crucial stage in achieving a new successful system and in giving confidence on the new system for the users that it will work effectively. The system can be implemented only after thorough testing is done and if it is found to work according to the specification. It involves careful planning, investigating of the current system and its constraints on implementation, design of methods to achieve the change over and an evaluation of change over methods apart from planning. The two major tasks of preparing implementation are education and training of the users and testing of the system.

Snapshots:

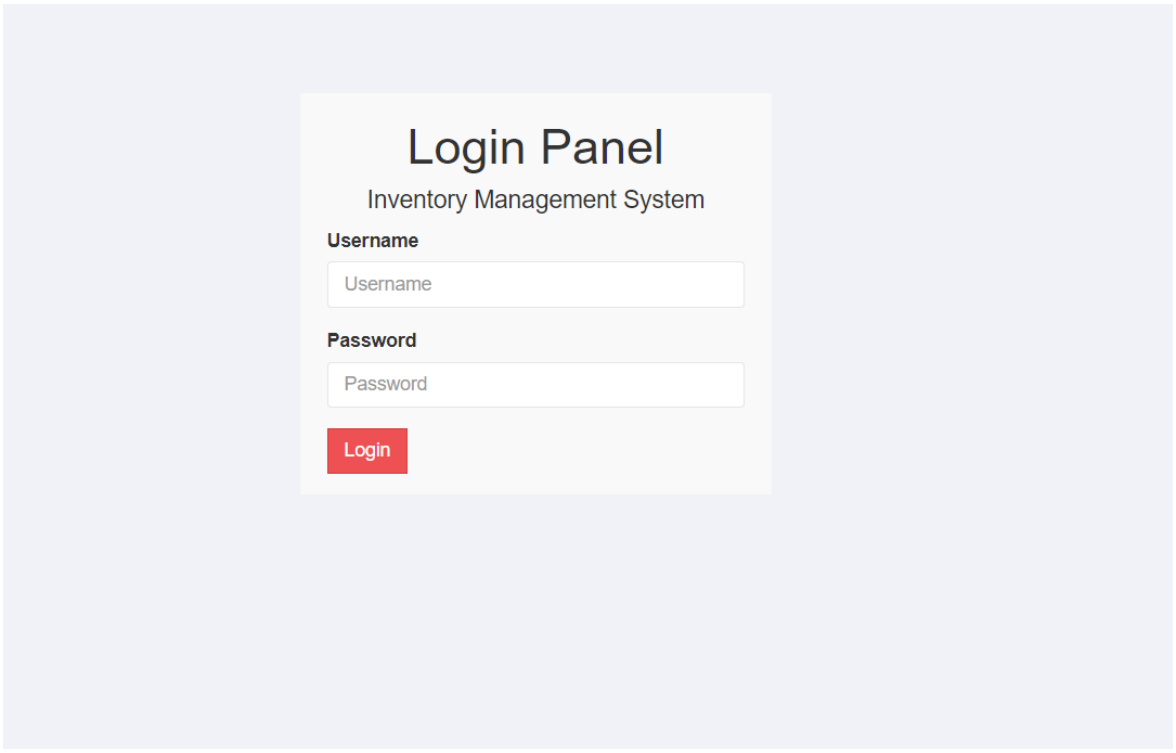
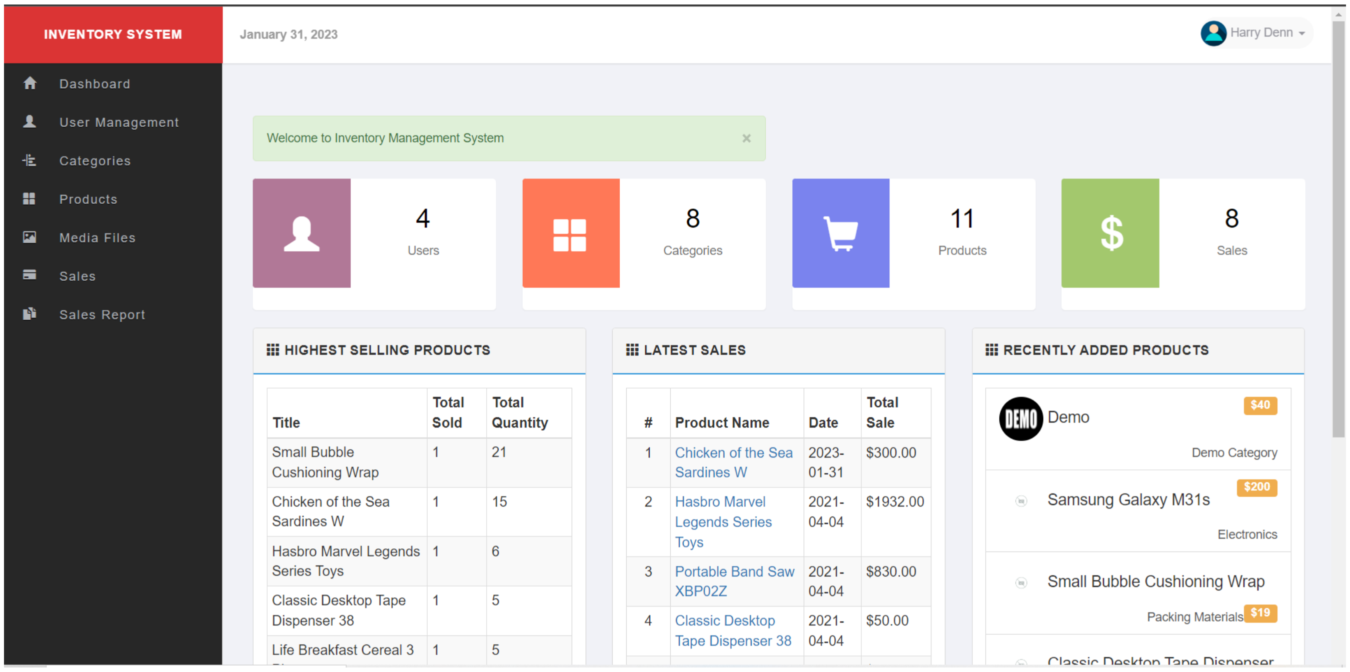


Figure 4.1 Login Page



Welcome to Inventory Management System

4 Users | 8 Categories | 11 Products | 8 Sales

HIGHEST SELLING PRODUCTS

Title	Total Sold	Total Quantity
Small Bubble Cushioning Wrap	1	21
Chicken of the Sea Sardines W	1	15
Hasbro Marvel Legends Series Toys	1	6
Classic Desktop Tape Dispenser 38	1	5
Life Breakfast Cereal 3	1	5

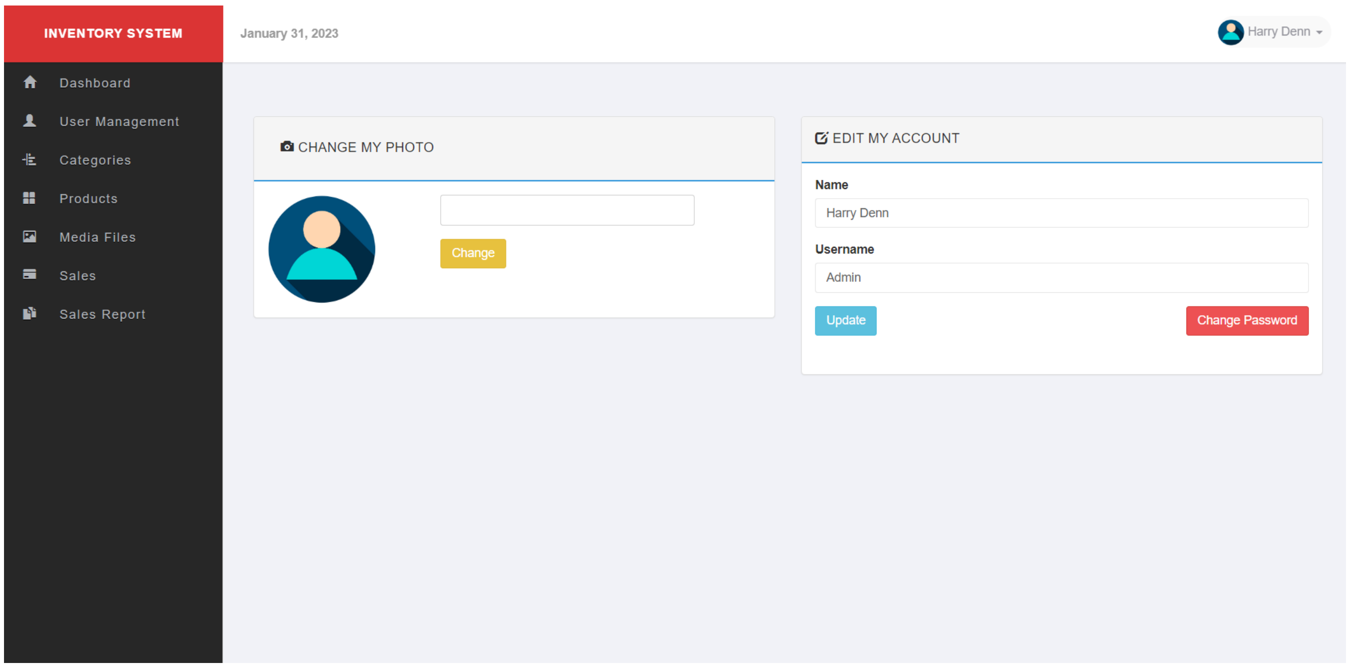
LATEST SALES

#	Product Name	Date	Total Sale
1	Chicken of the Sea Sardines W	2023-01-31	$300.00
2	Hasbro Marvel Legends Series Toys	2021-04-04	$1932.00
3	Portable Band Saw XBP02Z	2021-04-04	$830.00
4	Classic Desktop Tape Dispenser 38	2021-04-04	$50.00

RECENTLY ADDED PRODUCTS

Product	Category	Price
Demo	Demo Category	$40
Samsung Galaxy M31s	Electronics	$200
Small Bubble Cushioning Wrap	Packing Materials	$19

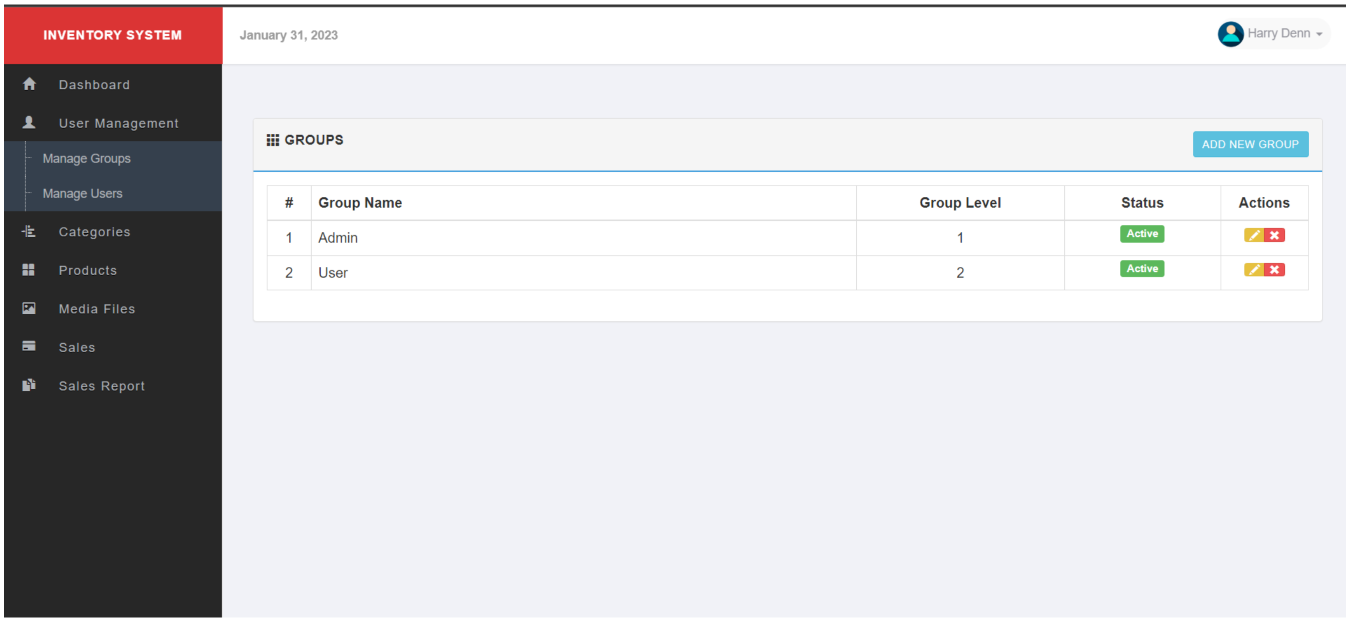
Figure 4.2 Dashboard view for admin



Name: Harry Denn

Username: Admin

Figure 4.3 Editing the profile of the user



#	Group Name	Group Level	Status	Actions
1	Admin	1	Active	
2	User	2	Active	

Figure 4.4 Managing groups of users

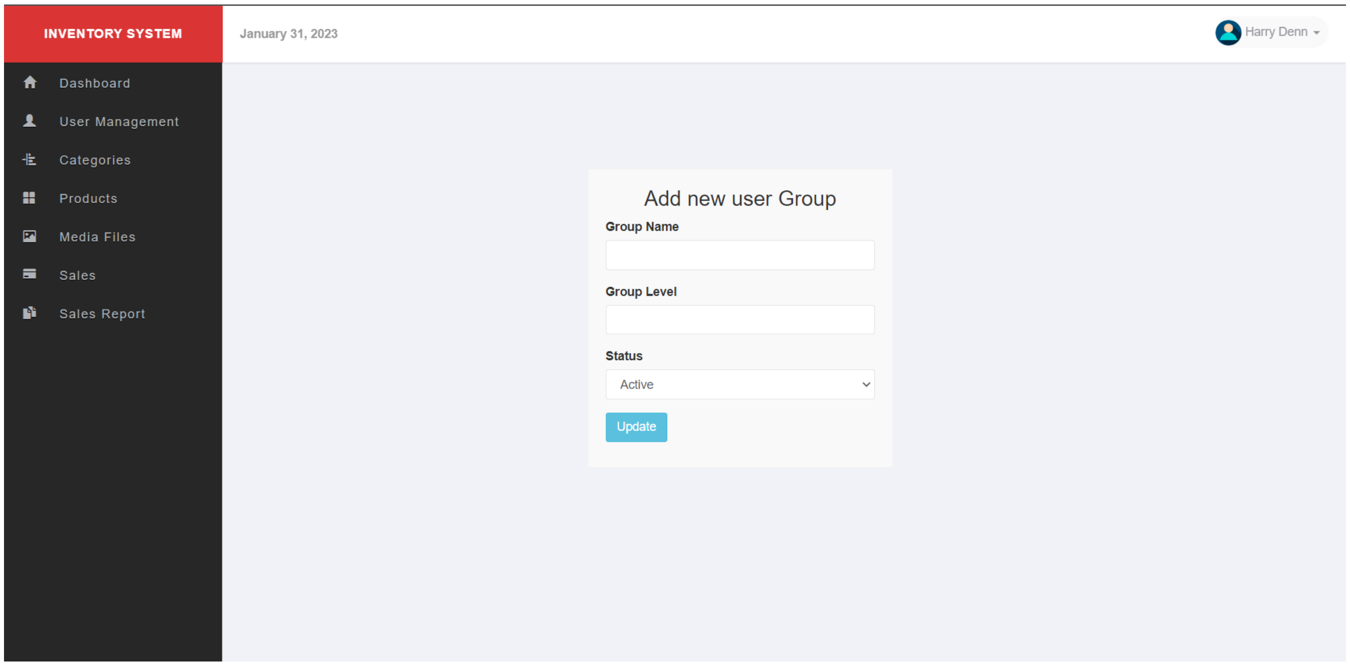
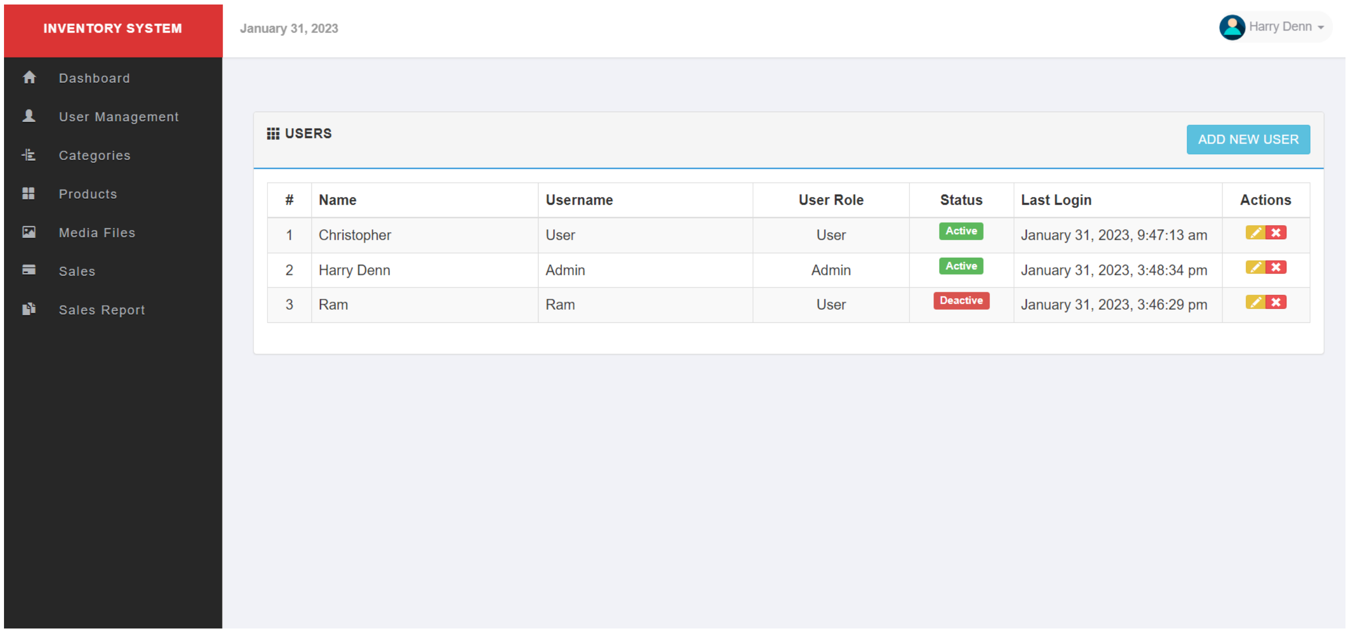


Figure 4.5 Adding new Group

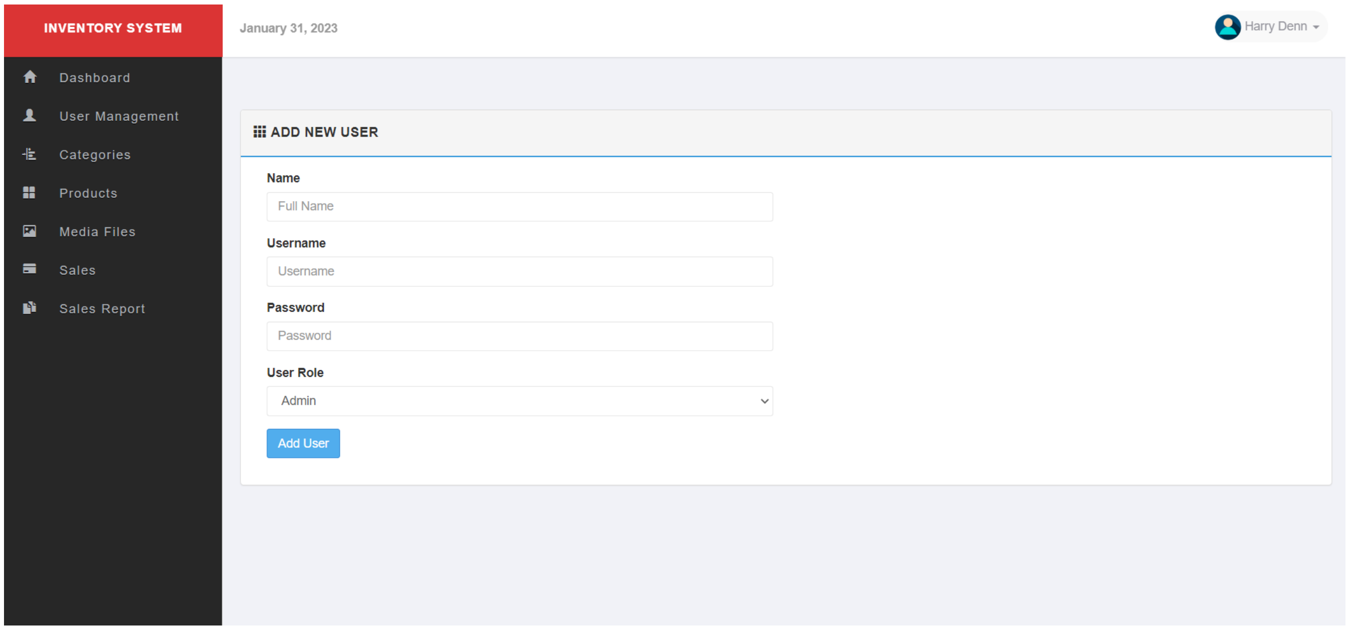


January 31, 2023

USERS

#	Name	Username	User Role	Status	Last Login	Actions
1	Christopher	User	User	Active	January 31, 2023, 9:47:13 am	
2	Harry Denn	Admin	Admin	Active	January 31, 2023, 3:48:34 pm	
3	Ram	Ram	User	Deactive	January 31, 2023, 3:46:29 pm	

Figure 4.6 Managing Users

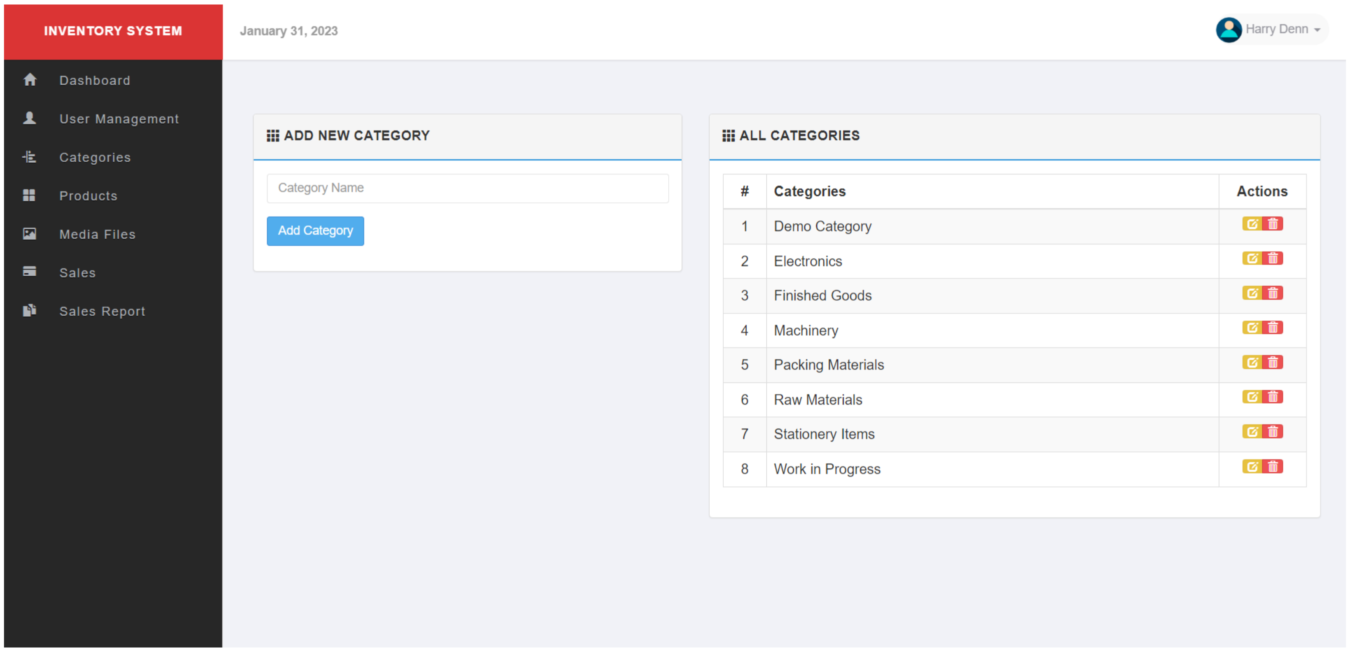


January 31, 2023

ADD NEW USER

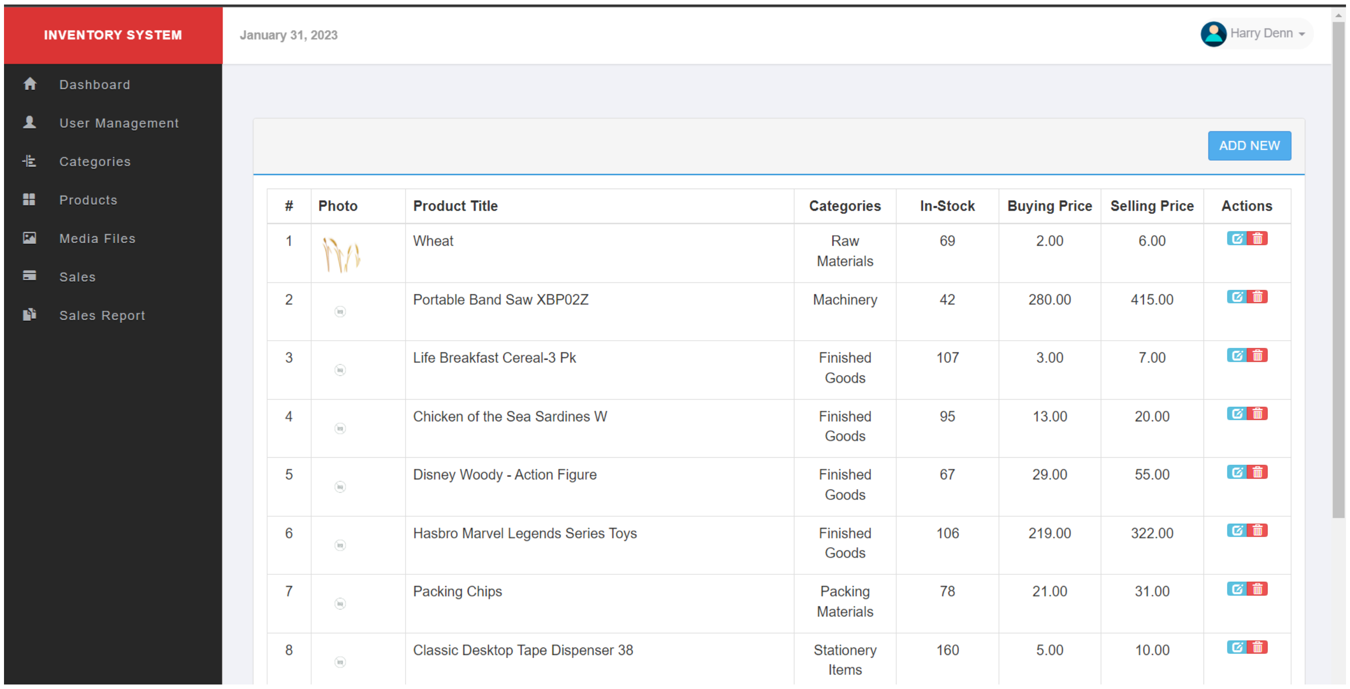
Name	Full Name
Username	Username
Password	Password
User Role	Admin

Figure 4.7 Adding new User



#	Categories	Actions
1	Demo Category	
2	Electronics	
3	Finished Goods	
4	Machinery	
5	Packing Materials	
6	Raw Materials	
7	Stationery Items	
8	Work in Progress	

Figure 4.8 Adding and Managing Category of Products



#	Photo	Product Title	Categories	In-Stock	Buying Price	Selling Price	Actions
1		Wheat	Raw Materials	69	2.00	6.00	
2		Portable Band Saw XBP02Z	Machinery	42	280.00	415.00	
3		Life Breakfast Cereal-3 Pk	Finished Goods	107	3.00	7.00	
4		Chicken of the Sea Sardines W	Finished Goods	95	13.00	20.00	
5		Disney Woody - Action Figure	Finished Goods	67	29.00	55.00	
6		Hasbro Marvel Legends Series Toys	Finished Goods	106	219.00	322.00	
7		Packing Chips	Packing Materials	78	21.00	31.00	
8		Classic Desktop Tape Dispenser 38	Stationery Items	160	5.00	10.00	

Figure 4.9 Managing products

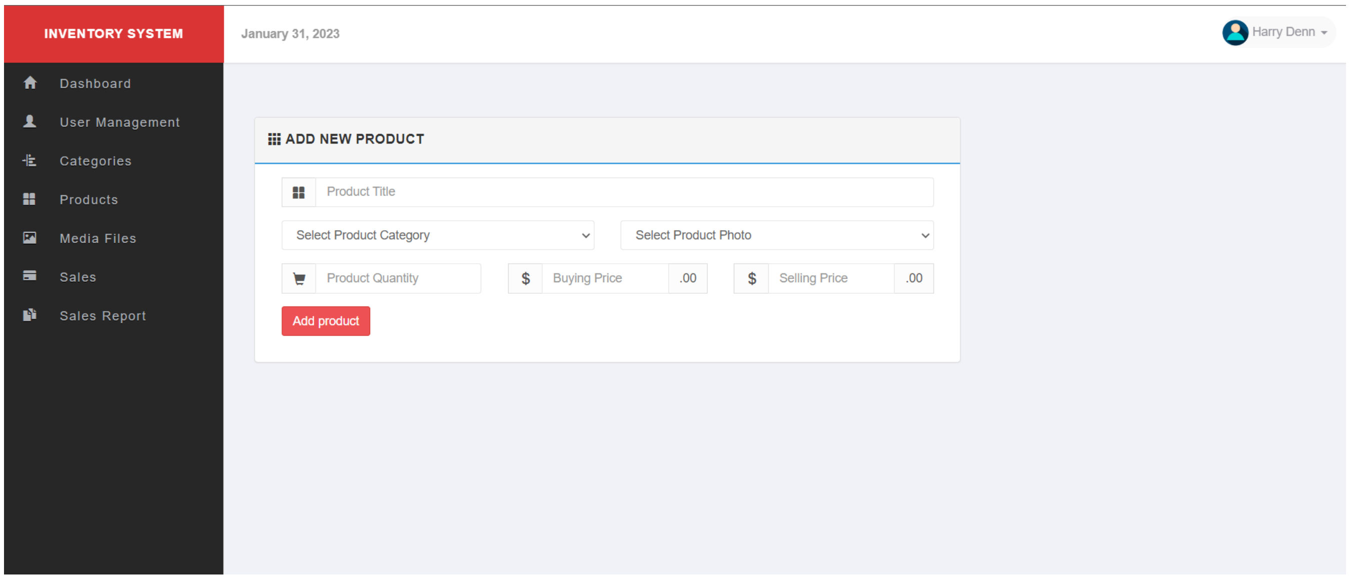
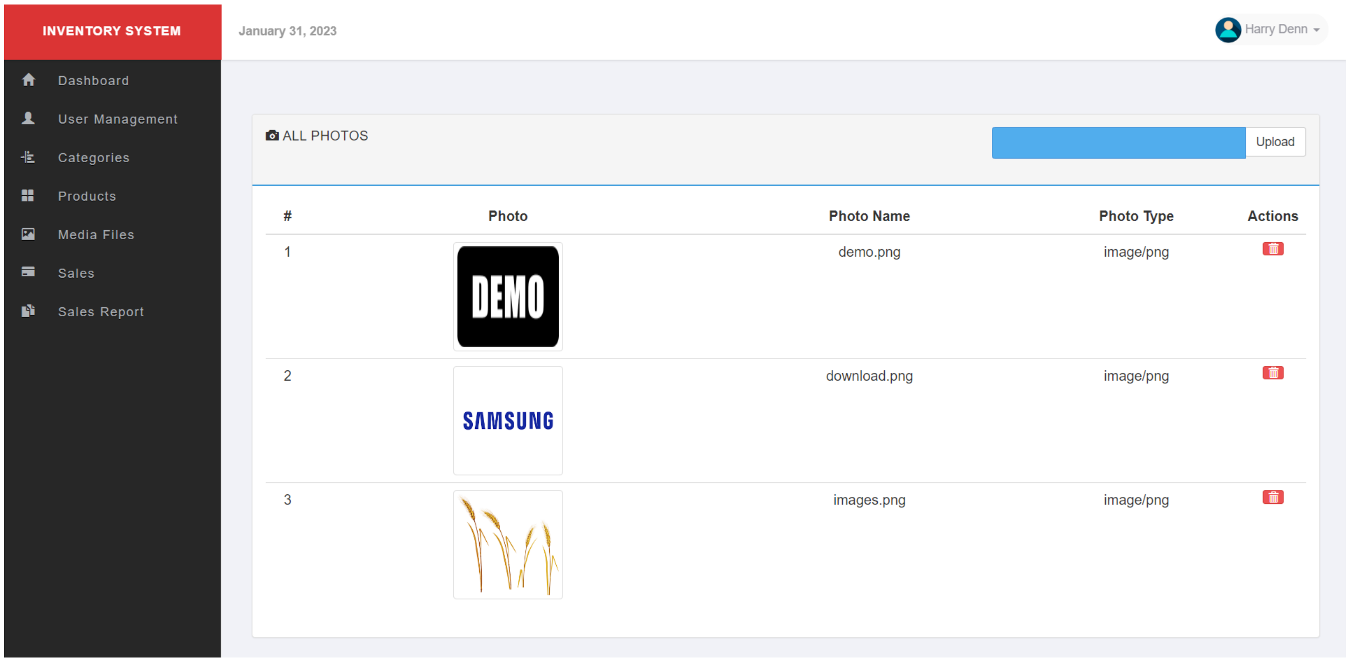
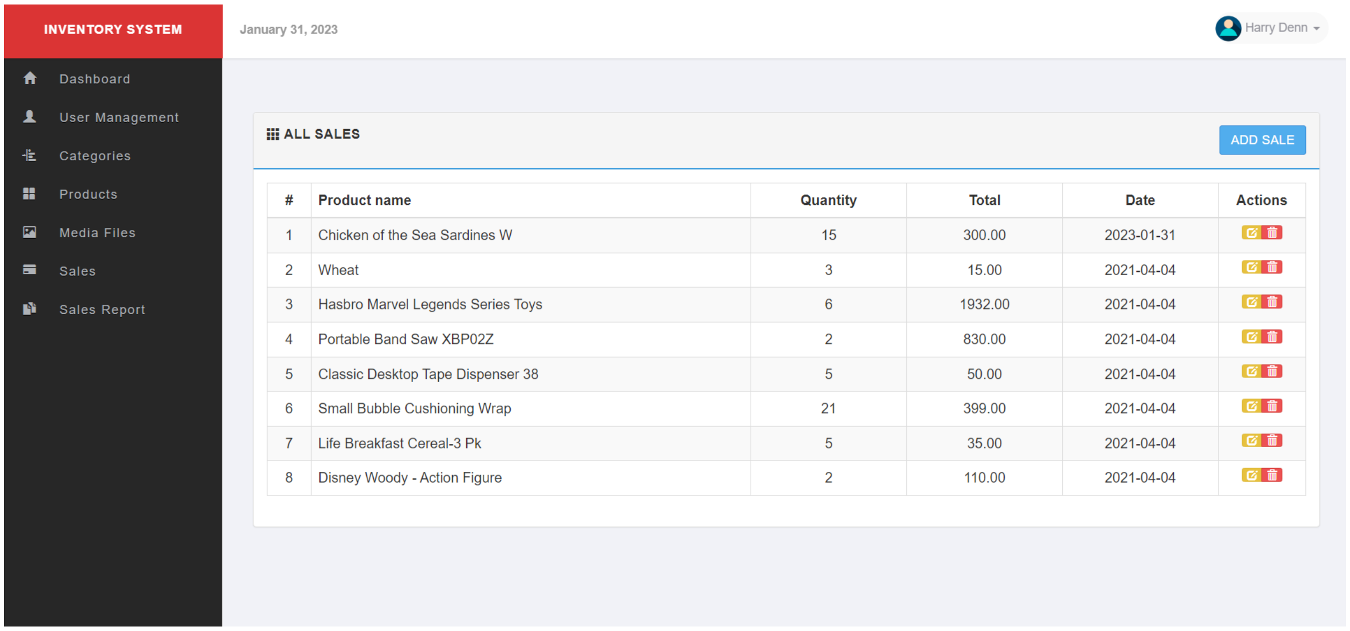


Figure 4.10 Adding new Product



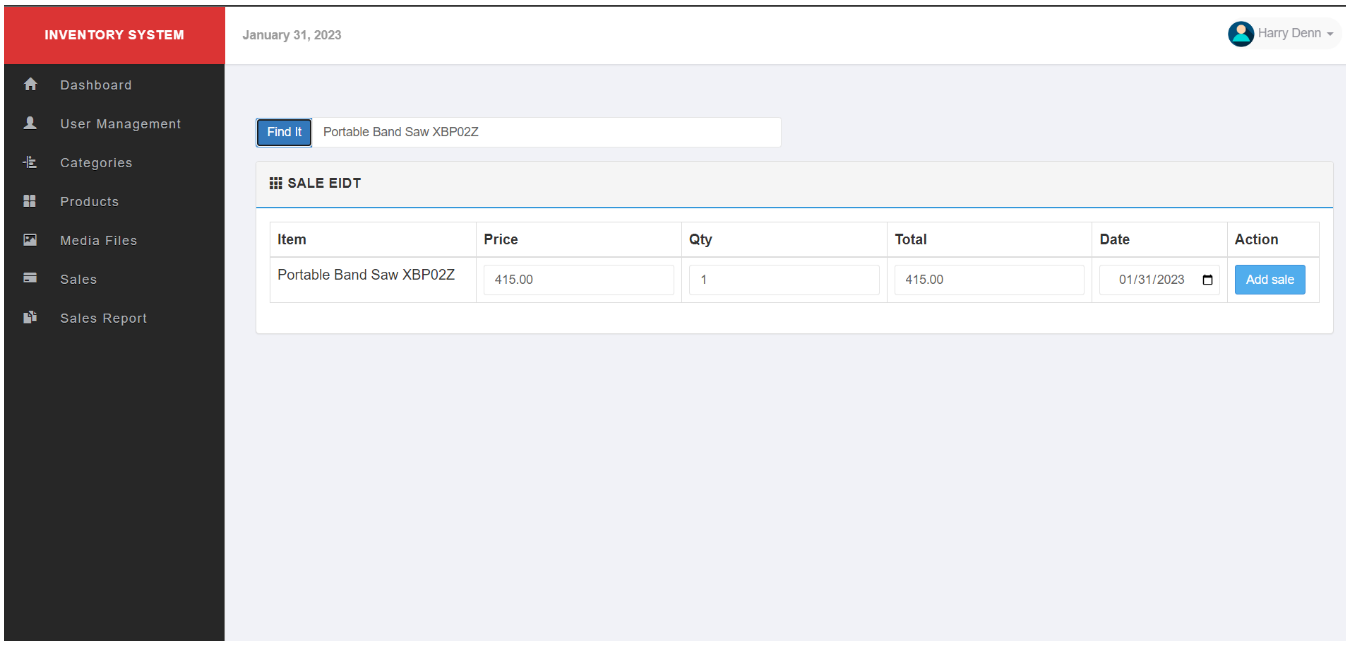
#	Photo	Photo Name	Photo Type	Actions
1	DEMO	demo.png	image/png	🗑
2	SAMSUNG	download.png	image/png	🗑
3		images.png	image/png	🗑

Figure 4.11 Adding and Managing Media Files



#	Product name	Quantity	Total	Date	Actions
1	Chicken of the Sea Sardines W	15	300.00	2023-01-31	
2	Wheat	3	15.00	2021-04-04	
3	Hasbro Marvel Legends Series Toys	6	1932.00	2021-04-04	
4	Portable Band Saw XBP02Z	2	830.00	2021-04-04	
5	Classic Desktop Tape Dispenser 38	5	50.00	2021-04-04	
6	Small Bubble Cushioning Wrap	21	399.00	2021-04-04	
7	Life Breakfast Cereal-3 Pk	5	35.00	2021-04-04	
8	Disney Woody - Action Figure	2	110.00	2021-04-04	

Figure 4.12 Managing Sales



Item	Price	Qty	Total	Date	Action
Portable Band Saw XBP02Z	415.00	1	415.00	01/31/2023	Add sale

Figure 4.13 Adding new Sales

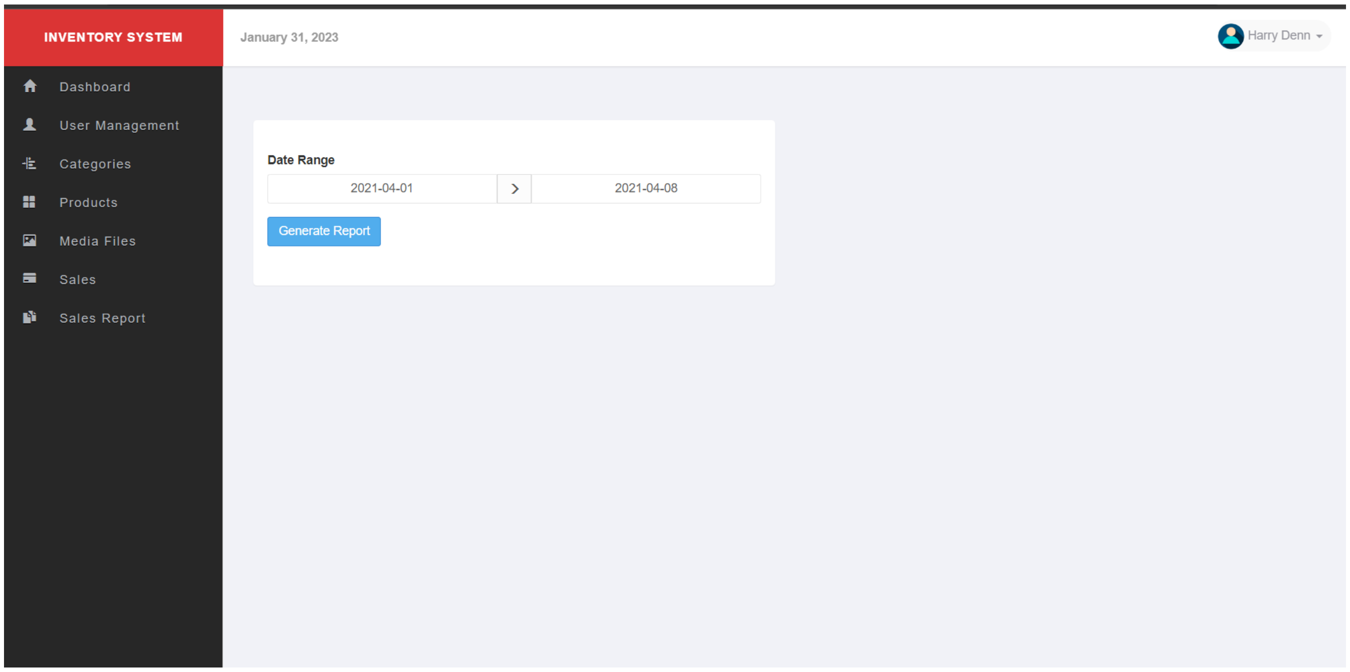
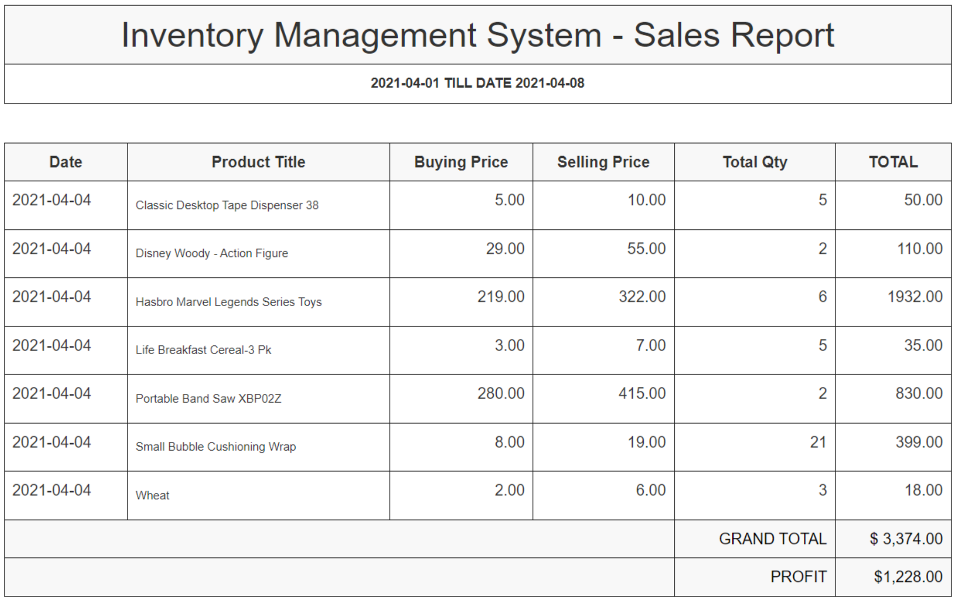


Figure 4.14 Setting date range for Sales Report

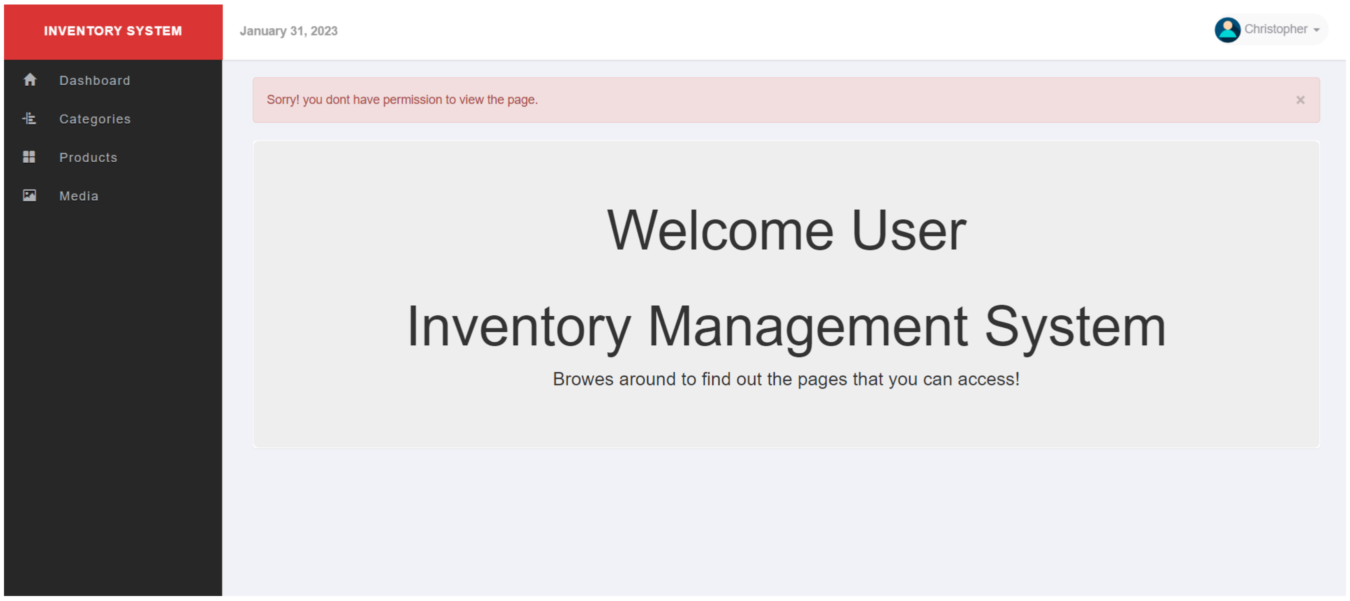


Inventory Management System - Sales Report

2021-04-01 TILL DATE 2021-04-08

Date	Product Title	Buying Price	Selling Price	Total Qty	TOTAL
2021-04-04	Classic Desktop Tape Dispenser 38	5.00	10.00	5	50.00
2021-04-04	Disney Woody - Action Figure	29.00	55.00	2	110.00
2021-04-04	Hasbro Marvel Legends Series Toys	219.00	322.00	6	1932.00
2021-04-04	Life Breakfast Cereal-3 Pk	3.00	7.00	5	35.00
2021-04-04	Portable Band Saw XBP02Z	280.00	415.00	2	830.00
2021-04-04	Small Bubble Cushioning Wrap	8.00	19.00	21	399.00
2021-04-04	Wheat	2.00	6.00	3	18.00
				GRAND TOTAL	$ 3,374.00
				PROFIT	$1,228.00

Figure 4.15 Generating Sales Report



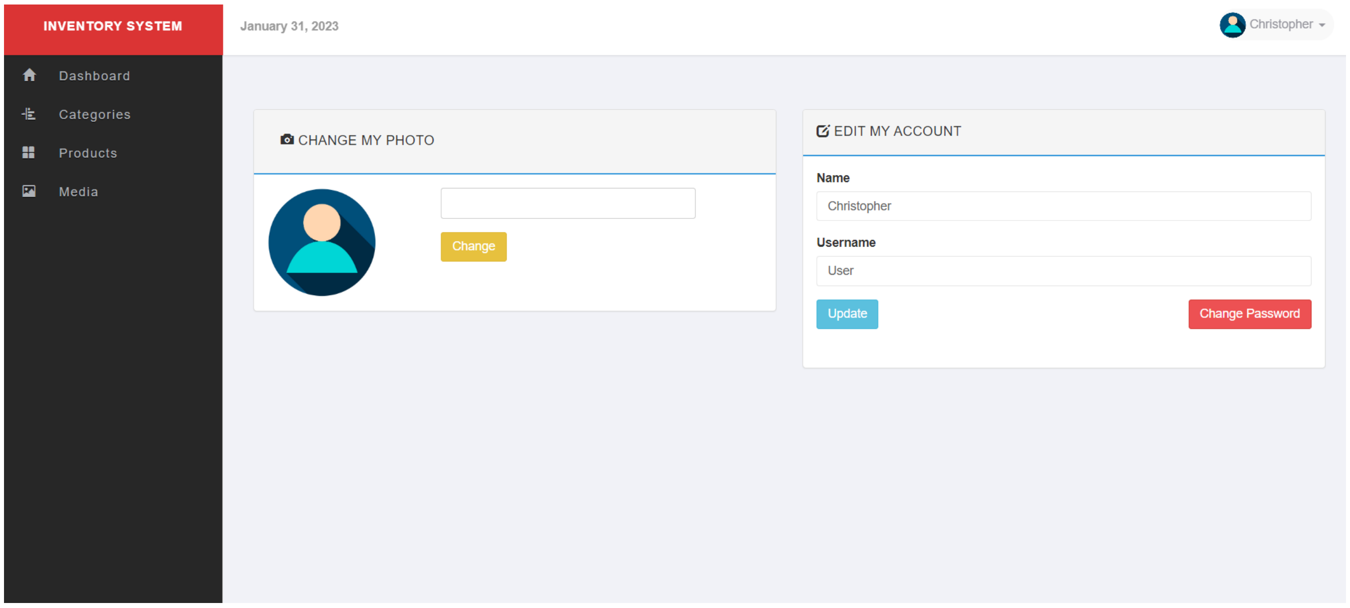
Sorry! you dont have permission to view the page.

Welcome User

Inventory Management System

Browes around to find out the pages that you can access!

Figure 4.16 Dashboard view for User



Name: Christopher

Username: User

Figure 4.17 User Settings for User

5 CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

It has been a great pleasure for to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in web based application and to some extent Windows Application and SQL Server, also about all handling procedure related with –Inventory Management. It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future.This will provide better opportunities and guidance in future in developing projects independently.

5.2 FUTURE ENHANCEMENTS

As we know India is a growing market and managing of products and inventory is very important. The movement of the raw materials or storing of them or transportation of them in a systematic manner is very important. It can also be the transportation of the finished good too. Each industry requires multiple products from different suppliers so to have a centralized system to hold these resources helps to improve the smooth operation. If the records of goods Is maintained well it can be beneficial to both the client and the seller to make profit/

The merits of thisproject are as follows:

- It’s a web-enabled project.
- This project offers user to enter the data through simple and interactive user interface. This is very helpfulfor the employee or admin to enter the desired information through so much simplicity.
- The user is mainly more concerned about the validity of the data, whatever he is entering. There arechecks on every stage of any new creation, data entry or updating so that the user cannot enter the invalid data, which can create problems at later date.
- Sometimes the user finds in the later stages of using project that he needs to update some of The information that he entered earlier. There are options for him by which he can update

the records.Moreover, there is restriction for his that he cannot change the primary data field. This keeps the validity of the data to longer extent.

- User is provided the option of monitoring the records he entered earlier. He can see the desiredrecords with the variety of options provided by him.
- From every part of the project the user is provided with the links through framing so that he can gofrom one option of the project to other as per the requirement. This is bound to be simple and very friendly as per the user is concerned. That is, we can say that the project is user friendly which is one of the primary concerns of any good project.
- Data storage and retrieval will become faster and easier to maintain because data is stored in asystematic manner and in a single database.
- Decision making process would be greatly enhanced because of faster processing of informationsince data collection from information available on computer takes much less time then manual system.

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- Going through products and their availability as and where we are.
- More details about a product can be added.
- Tutorial videos of how to use the products can be added
- Bar code scanning for products can be added
- Online paying system can be integrated to the system
- The platform can be hosted on the online servers to make it accessible worldwide.
- Integrate multiple load balancers to distribute the loads of the system.
- Implement a backup mechanism for taking backup of database on regular basis on different servers.

The above-mentioned points are the enhancements which can be done to increase the applicability and usage of this project. Through these features it will increase the efficiency, accuracy and transparency.

6 REFERENCES

1. Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7thEdition, 2017, Pearson.

2. Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGrawHill