## CHAPTER 1

**INTRODUCTION**

### Introduction to Database Management System

A database management system (DBMS) refers to the technology for creating and managing databases. DBMS is a software tool to organize (create, retrieve, update, and manage) data in a database. A software for storing and retrieving user by considering appropriate security measures. It allows users to create their own databases as per their requirement.

It consists of group of programs which manipulate the database and provide an interface between the database. It includes the user of the database and other application programs. The DBMS accepts the request for data from an application and instructs the operating system to provide the specific data. In large systems, a DBMS helps users and other third-party software to store and retrieve data.

If it is any field that has contributed to the greatest advancements in the world today, it is the field of information technology, commonly known by the acronym IT. When most people hear of the phrase ‘information technology’, they tend to picture several Tecno-geeks seated by the computer and fixing some computer software.

### Types of DBMS

There are mainly 4 types of DBMS, which are Hierarchical, Relational, Network, and Object-oriented DBMS

* **Hierarchical DBMS**: As the name suggests, this type of DBMS has a style of predecessor- successor type of relationship. So it has a structure similar to that of a tree, wherein the nodes represent records and the branches of the tree represent fields.
* **Relational DBMS (RDBMS)**: This type of DBMS , uses a structure that allows the users to identify and access data in relation to another piece of data in the database.
* **Network DBMS**: This type of DBMS supports many to many relations wherein multiple member records can be linked.
* **Object-oriented DBMS**: This type of DBMS uses small individual software called objects. Each object contains a piece of data, and the instructions for the actions to be done with the data.

### Applications of DBMS

* + 1. **Library Management System** : There are lots of books in the library so; it is tough to store the record of all the books in a register or copy. So, the database management system (DBMS) is used to maintain all the information related to the name of the book, issue date, availability of the book, and its author.
    2. **Railway Reservation System** : In the railway reservation system, the database is required to store the record or data of ticket bookings, status about train’s arrival, and departure. Also, if trains get late, people get to know it through database update.
    3. **Banking** : Database management system is used to store the transaction information of the customer in the database.
    4. **Credit card transactions** : Database Management system is used for purchasing on credit cards and generation of monthly statements.
    5. **Education Sector** : Presently, examinations are conducted online by many colleges and universities. They manage all examination data through the database management system (DBMS). In spite that student’s registrations details, grades, courses, fee, attendance, results, etc. all the information is stored in the database.
    6. **Social Media Sites** : We all use of social media websites to connect with friends and to share our views with the world. Daily, millions of peoples sign up for these social

media accounts like Pinterest, Facebook, Twitter, and Google plus. By the use of the database management system, all the information of users are stored in the database and, we become able to connect with other people.

* + 1. **Telecommunications** : Without DBMS any telecommunication company can’t think. The Database management system is necessary for these companies to store the call details and monthly postpaid bills in the database.
    2. **Finance** : The database management system is used for storing information about sales, holding and purchases of financial instruments such as stocks and bonds in a database.
    3. **Online Shopping** : Everyone wants to shop through online shopping websites (such as Amazon, Flipkart, snap deal) from home. So, all the products are sold and added only with the help of the database management system (DBMS). Invoice bills, payments, purchase information all of these are done with the help of DBMS.
    4. **Human Resource Management** : Big firms or companies have many workers or employees working under them. They store information about employee’s salary, tax, and work with the help of database management system (DBMS).
    5. **Manufacturing** : Manufacturing companies make different types of products and sale them on a daily basis. In order to keep the information about their products like bills, purchase of the product, quantity, supply chain management, database management system (DBMS) is used.
    6. **Airline Reservation System** : This system is the same as the railway reservation system. This system also uses a database management system to store the records of flights departure, arrival, and delay status.

### Introduction to MySQL

MySQL is a Relational Database Management System (“RDBMS”). It is used by most modern websites and web-based services as a convenient and fast-access storage and retrieval solution for large volumes of data. MySQL is open-source and free software under the GNU license. It is supported by Oracle Company.

It is developed, marketed, and supported by MySQL AB, a Swedish company, and written in C programming language and C++ programming language. MySQL supports many Operating Systems like Windows, Linux, MacOS, etc. with C, C++, and Java languages.

MySQL can also be accessed using many tools. It can be easily communicated with via PHP (PHP Hypertext Preprocessor), a scripting language whose primary focus is to manipulate HTML for a webpage on the server before it is delivered to a client’s machine. A user can submit queries to a database via PHP, allowing insertion, retrieval and manipulation of information into/from the database.

MySQL server design is multi-layered with independent modules and is fully multithreaded by using kernel threads. It can handle multiple CPUs if they are available. MySQL Server works in client/server or embedded systems. and it works on many different platforms.

### MySQL Command Syntax

The four main categories of SQL statements are as follows

#### DML (Data Manipulation Language) :

DML statements affect records in a table. These are basic operations we perform on data such as selecting a few records from a table, inserting new records, deleting unnecessary records, and updating/modifying existing records

* + - * **SELECT** – select records from a table

#### SELECT column1, column2, ... FROM table\_name;

* + - * **INSERT** – insert new records

#### INSERT INTO table\_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);

* + - * **UPDATE** – update/Modify existing records

#### UPDATE table\_name

**SET column1 = value1, column2 = value2, ... WHERE condition;**

* + - * **DELETE** – delete existing records

#### DELETE FROM table\_name WHERE condition;

#### DDL (Data Definition Language)

DDL statements are used to alter/modify a database or table structure and schema. These statements handle the design and storage of database objects.

* + - * **CREATE** – create a new Table, database, schema **CREATE TABLE table\_name ( column1 datatype,**

#### column2 datatype, column3 datatype, ....

**);**

* + - * **ALTER** – alter existing table, column description

#### ALTER TABLE table\_name ADD column\_name datatype;

* + - * **DROP** – delete existing objects from database

#### DROP TABLE table\_name;

#### DCL (Data Control Language)

DCL statements control the level of access that users have on database objects.

* + - * **GRANT** – allows users to read/write on certain database objects **GRANT ALL PRIVILEGES ON database\_name.\* TO 'username'@'localhost';**
      * **REVOKE** – keeps users from read/write permission on database objects

#### REVOKE privileges ON object FROM user;

#### TCL (Transaction Control Language)

TCL statements allow you to control and manage transactions to maintain the integrity of data within SQL statements.

* + - * **BEGIN** Transaction – opens a transaction **[begin\_label:] BEGIN [statement\_list]**

#### END [end\_label]

* + - * **COMMIT** Transaction – commits a transaction

#### COMMIT [ { TRAN | TRANSACTION } [ transaction\_name | @tran\_name\_variable ] ] [ WITH ( DELAYED\_DURABILITY = { OFF | ON } ) ]

**[ ; ]**

* + - * **ROLLBACK** Transaction – ROLLBACK a transaction in case of any error

#### ROLLBACK { TRAN | TRANSACTION }

**[ transaction\_name | @tran\_name\_variable**

**| savepoint\_name | @savepoint\_variable ] [ ; ]**

## CHAPTER 2

**ANALYSIS AND REQUIREMENT SPECIFICATION**

### Purpose of this project

The Clothing Shop Management System aims to provide a user-friendly platform for efficiently managing clothing inventory and sales processes. By automating tasks that were previously done manually, such as inventory tracking and order processing, the system significantly reduces the time and effort required to manage the shop's operations. With the increasing demand for clothing and the growing customer base, an automated system becomes crucial to meet the needs of customers in a timely manner. Utilizing programming and database techniques, the system streamlines the process of purchasing and managing clothing items online, offering customers the convenience of browsing, purchasing, and managing their orders from anywhere, at any time. Ultimately, the goal of the Clothing Shop Management System is to simplify the shopping experience for customers while enhancing the overall efficiency and effectiveness of the shop's operations.

### Scope of this project

* Inventory Management: The system will efficiently manage the inventory of clothing items, including tracking stock levels, adding new items, updating existing items, and removing items when necessary. It ensures that the clothing shop always has the right products available to meet customer demand.
* Sales and Billing: The system will handle the sales process, allowing staff to generate bills for customers, calculate total amounts including taxes, and process various payment methods such as cash, credit cards, or digital wallets. It ensures accurate and timely billing for all transactions.
* Customer Management: The system will maintain a database of customer information, including profiles, purchase history, preferences, and contact details. It enables personalized interactions with customers, such as sending promotions, managing loyalty programs, and providing excellent customer service.

### Functional Requirements

#### Modules

* Manager Management Module: Allows managing manager accounts, including creation, updating, and deletion, as well as role assignment.
* Inventory Management Module:Facilitates adding, updating, and monitoring clothing inventory, including product details and stock levels.
* Offers Management Module:Enables creating, updating, and deleting special offers and discounts for clothing products.
* Billing Management Module: Handles generating bills, calculating total amounts, and recording sales transactions for clothing purchases.
* Feedback Management Module: Manages capturing and storing customer feedback and ratings for clothing products and the shopping experience.

### Non-Functional Requirements

#### Hardware requirements

* + - * **Processor** – Intel Core i3, 2Ghz or above for PCs and 1Ghz
      * **Memory Space** – 1GB recommended.
      * **RAM** – 2GB or above.

#### Software requirements

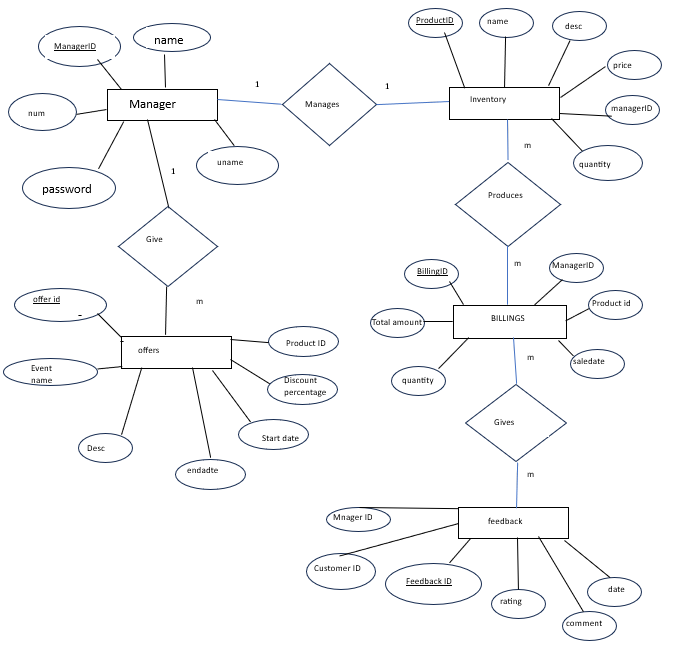
* + - * **Operating System** – Windows 7 and above, MacOS X and above
      * **Browser** – Google Chrome, Microsoft Edge or any other basic browsers.
      * **Front End** – HTML, CSS, JS
      * **Back End** – MySQL
      * **Serverside Scripting** – PHP

## CHAPTER

**DESIGN**

### Entity Relationship Diagram

Data schema in graphical form is called ER Diagram. It is usually drawn in a graphical form as boxes (entities) that are connected by lines (relationships) which express the associations and dependencies between entities. An entity–relationship model is usually the result of systematic analysis to define and describe what is important to processes in an area of a business. An ER model does not define the business processes; it only presents a business.



N

N

N

Fig 3.1 ER diagram for cloth shop database

### Schema Diagram

The term "schema" refers to the organization of data as a blueprint of how the database is constructed. The formal definition of a database schema is a set of formulas called integrity constraints imposed on a database. A relational schema shows references among fields in the database. When a primary key is referenced in another table in the database, it is called a foreign key. This is denoted by an arrow with the head pointing at the referenced key attribute. A schema diagram helps organize values in the database. The following diagram shows the schema diagram for the database.

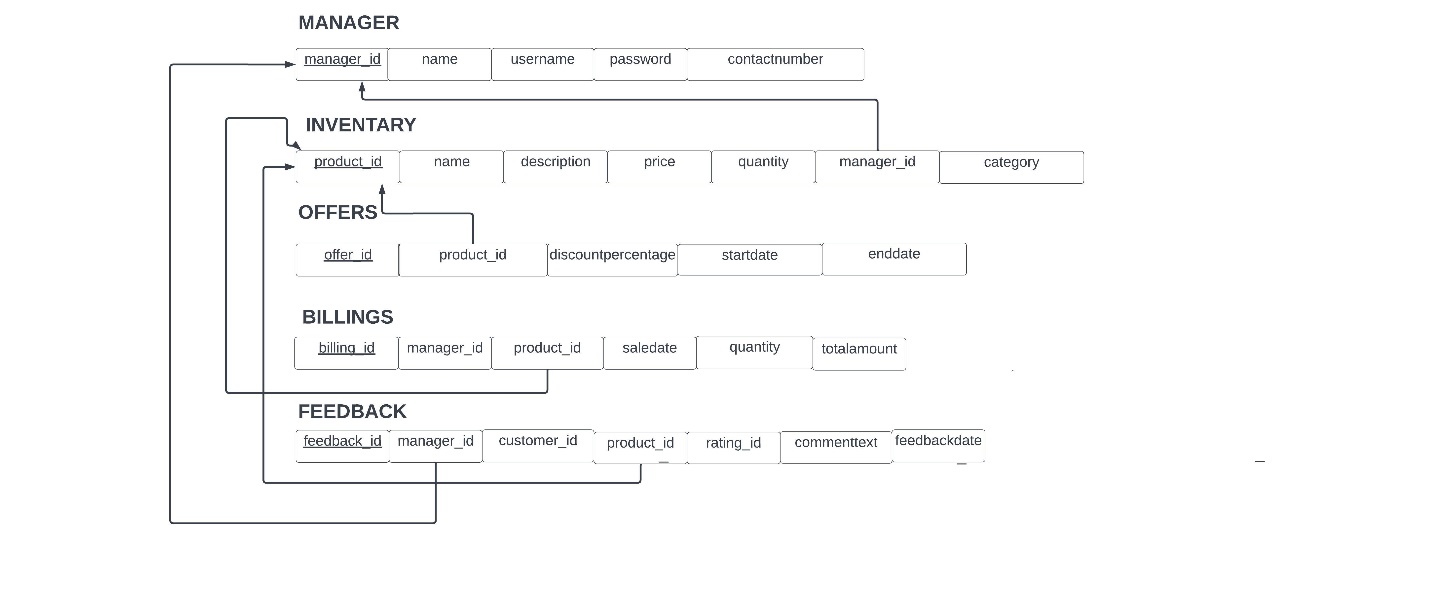


Fig 3.2 Schema diagram of cloth shop database

## CHAPTER 4

**IMPLEMENTATION**

### 4.1 Implementation of Table Creation page

* Manager Table

CREATE TABLE Manager (

ManagerID INT PRIMARY KEY,

Name VARCHAR(255),

Username VARCHAR(50),

Password VARCHAR(50),

ContactNumber VARCHAR(15)

);

* + Inventory Table

CREATE TABLE Inventory (

ProductID INT PRIMARY KEY,

Name VARCHAR(255),

Description TEXT,

Price DECIMAL(10, 2),

Quantity INT,

ManagerID INT,

Category ENUM('Men', 'Women', 'Child'), -- Category can be Men, Women, or Child

FOREIGN KEY (ManagerID) REFERENCES Manager(ManagerID)

);

* + Offers Table

CREATE TABLE Offers (

OfferID INT PRIMARY KEY,

ProductID INT,

DiscountPercentage DECIMAL(5, 2),

StartDate DATE,

EndDate DATE,

FOREIGN KEY (ProductID) REFERENCES Inventory(ProductID)

);

* + Billings Table

CREATE TABLE Billings (

BillingID INT PRIMARY KEY,

ManagerID INT,

ProductID INT,

SaleDate DATE,

Quantity INT,

TotalAmount DECIMAL(10, 2),

FOREIGN KEY (ManagerID) REFERENCES Manager(ManagerID),

FOREIGN KEY (ProductID) REFERENCES Inventory(ProductID),

INDEX (ManagerID, ProductID)

);

* + Feedback Table

CREATE TABLE Feedback (

FeedbackID INT PRIMARY KEY,

ManagerID INT,

CustomerID INT,

ProductID INT,

Rating INT,

Comment TEXT,

FeedbackDate DATE,

FOREIGN KEY (ManagerID) REFERENCES Manager(ManagerID),

FOREIGN KEY (ProductID) REFERENCES Inventory(ProductID)

);

**4.2 Implementation of Connection:**

<?php

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "cloth";

// Create connection

$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

?>

**4.3 Implementation of Updation**

<?php

include("database.php");

// Check if form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Get values from the form

$productID = $\_POST["ProductID"];

$name = $\_POST["Name"];

$description = $\_POST["Description"];

$price = $\_POST["Price"];

$quantity = $\_POST["Quantity"];

$managerID = $\_POST["ManagerID"];

$category = $\_POST["Category"];

// Update query

$sql = "UPDATE Inventory SET

Name = '$name',

Description = '$description',

Price = $price,

Quantity = $quantity,

ManagerID = $managerID,

Category = '$category'

WHERE ProductID = $productID";

// Execute the query

if ($conn->query($sql) === TRUE) {

echo "Record updated successfully";

} else {

echo "Error updating record: " . $conn->error;

}

// Close the database connection

$conn->close();

}

?>

**4.3 Implementation of Insertion:**

<?php

include("database.php"); // Assuming you have a database connection file

// Check if the form is submitted

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Collect form data

$productID = $\_POST["ProductID"];

$name = $\_POST["Name"];

$description = $\_POST["Description"];

$price = $\_POST["Price"];

$quantity = $\_POST["Quantity"];

$managerID = $\_POST["ManagerID"];

$category = $\_POST["Category"];

// SQL to insert a new record

$insertSQL = "INSERT INTO Inventory (ProductID, Name, Description, Price, Quantity, ManagerID, Category)

VALUES ($productID, '$name', '$description', $price, $quantity, $managerID, '$category')";

// Execute the query

if ($conn->query($insertSQL) === TRUE) {

echo "New record has been inserted successfully.";

} else {

echo "Error inserting record: " . $conn->error;

}

}

// Close the database connection

$conn->close();

?>

**4.4 Implementation of Search Module:**

<script>

function searchInventory() {

var input, filter, table, tr, td, i, txtValue;

input = document.getElementById("inventorySearchInput");

filter = input.value.toUpperCase();

table = document.querySelector("table");

tr = table.getElementsByTagName("tr");

for (i = 1; i < tr.length; i++) {

td = tr[i].getElementsByTagName("td")[1];

if (td) {

txtValue = td.textContent || td.innerText;

if (txtValue.toUpperCase().indexOf(filter) > -1) {

tr[i].style.display = "";

} else {

tr[i].style.display = "none";

}

}

}

}

</script>

**4.5 Implementation of Deletion:**

<?php

include("database.php"); // Assuming you have a database connection file

if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

// Check if ProductID is provided

if (isset($\_POST["ProductID"]) && !empty($\_POST["ProductID"])) {

$productID = $\_POST["ProductID"];

// SQL to delete the record based on ProductID

$deleteSQL = "DELETE FROM Inventory WHERE ProductID = $productID";

// Execute the query

if ($conn->query($deleteSQL) === TRUE) {

echo "Record with ProductID $productID has been deleted successfully.";

} else {

echo "Error deleting record: " . $conn->error;

}

} else {

echo "ProductID is required.";

}

}

// Close the database connection

$conn->close();

?>

## CHAPTER 5

**SNAPSHOTS**

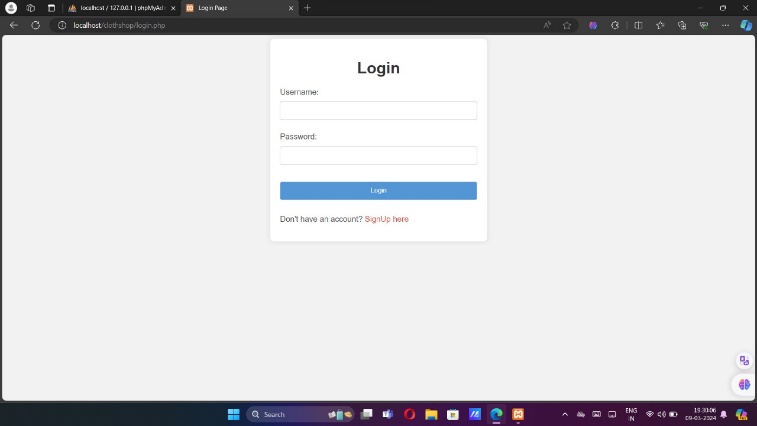
****

fig 5.2 Login Page

Manager Login Page : here Manager can Login Using Our Login Credentials

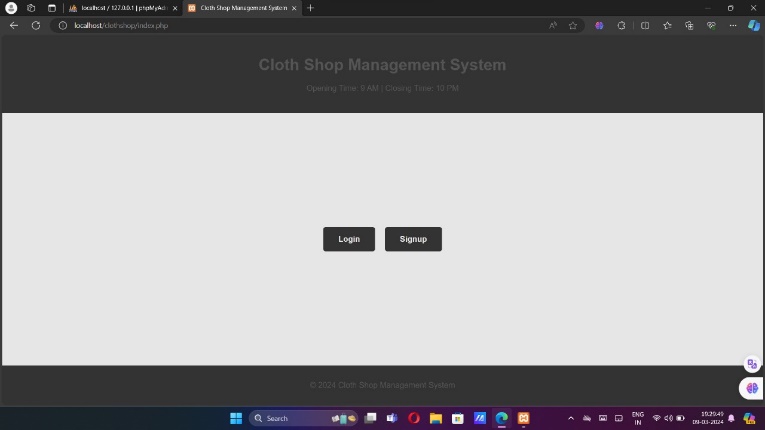
****

fig 5.2 Home Page

Home page Which has opening and closing time of the Shop along with that it also has the login option for manager

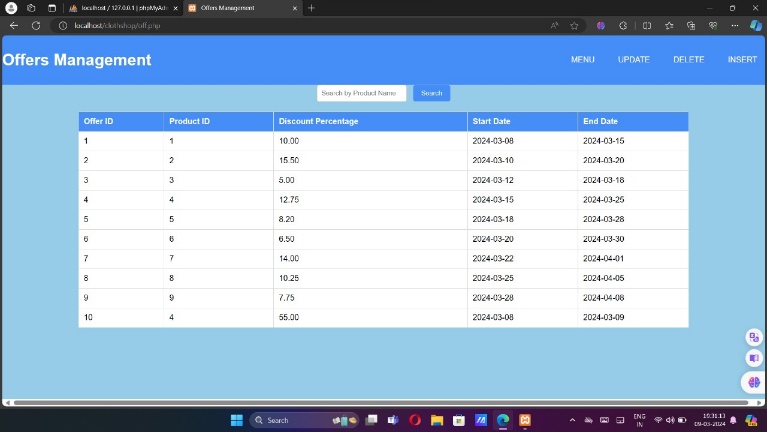


Fig 5.3 Retrieving from the table

Retrieving Data of Inventory Table: Manager can see all retrieved data and also Have search Button if many Entries are there.

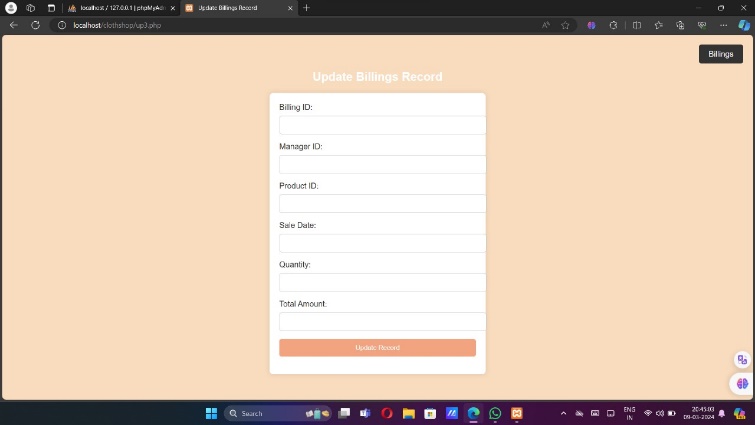


Fig 5.4 Update options

Updating Data to the Table: We can update The Data To the table. Tables are: Sales, Inventory, Vehicle Booking

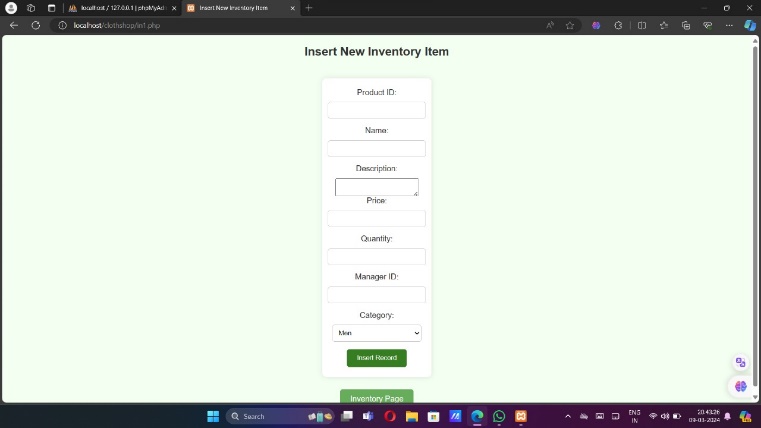


Fig 5.5 Inserting into the table

Inserting Data to the Table: We can insert The Data To the table. Tables are: Sales, Inventory, Vehicle Booking

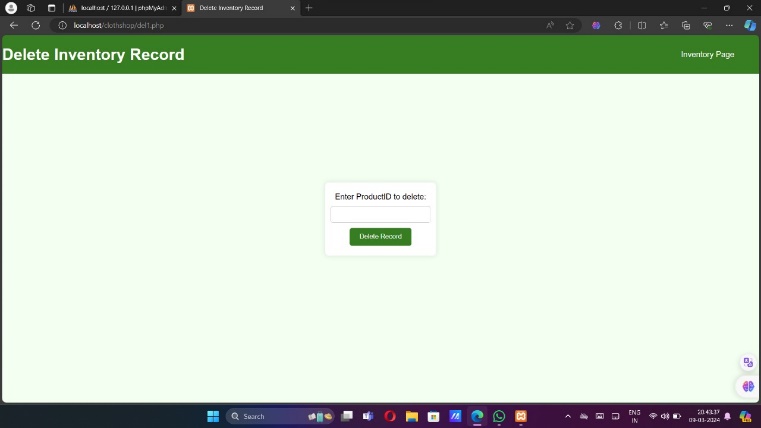


Fig 5.6 Deleting from the table

Updating Data to the Table: We can update The Data To the table. Tables are: Sales, Inventory, Vehicle Booking

## CHAPTER 6

**CONCLUSION**

In conclusion, database management systems (DBMS) play a crucial role in various sectors such as education, finance, telecommunications, and more, by efficiently organizing and managing data. MySQL, as a relational database management system, offers a robust solution for storing and retrieving data, supporting modern websites and web-based services. With its comprehensive command syntax and functionality, MySQL facilitates the implementation of projects like the one outlined here, which aims to streamline processes and enhance user experiences.

## REFERENCES

1. Fundamentals of Database Systems, Ramez Elmasri and Shamkant B. Navathe, 7th Edition, 2017, Pearson.
2. Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill.