### **Inventory Management System Documentation**

**Cover Page**

**Project Title**: Inventory Management System  
**Team Name**: Team 5  
**Date of Submission**: 28/02/2025

**Team Members**:

* Nihan P
* Devadath Gireesh
* Ajhin KJ
* Jithin VM
* Mariyam Mubbena

### **Acknowledgment**

We would like to express our sincere gratitude to [Insert Name] for their guidance and support throughout this project. We would also like to thank our teammates for their collaboration and hard work in successfully completing the project. Special thanks to the institution for providing the necessary resources to bring this project to life.

### **Abstract**

The Inventory Management System is designed to help businesses efficiently manage their product inventory by allowing users to add, delete, and update products as well as track available stock. The system is built using Python and MySQL for database management. This project automates the process of inventory tracking, ensuring that businesses can manage product stock levels in real-time with minimal errors. The expected outcome is a fully functional inventory system that enhances operational efficiency and reduces manual tracking errors.

### **Introduction**

* **Overview of the Project**:  
  The Inventory Management System is a software solution designed to manage the inventory of products for small and medium-sized businesses. The system offers basic functionalities such as adding new products, deleting existing products, updating product quantities, and viewing the available stock of products in the database. It ensures real-time data accuracy and simplifies inventory management tasks.
* **Purpose and Importance**:  
  The primary goal of this system is to automate and streamline inventory management for businesses, reducing the manual effort required to track stock levels. It ensures accurate stock management, prevents overstocking or understocking, and provides a quick overview of product quantities.
* **Problem Statement**:  
  Inventory management can be error-prone and inefficient when done manually. Businesses often face challenges such as discrepancies in stock levels, leading to excess stock or stockouts. This project aims to provide an automated system that addresses these issues by offering a streamlined, accurate inventory management solution.
* **Objectives**:
  + To develop an easy-to-use system for managing product inventory.
  + To enable users to add, delete, and update products in the system.
  + To track and view available stock in real-time.

### **Literature Review / Background Study**

* **Existing Solutions or Related Work**:  
  Several inventory management systems, such as SAP, Zoho Inventory, and TradeGecko, provide extensive features for businesses to manage inventory. These systems often include advanced reporting, multi-location management, and integration with other business functions like sales and accounting.
* **Gaps in Existing Systems**:  
  While large-scale systems offer many advanced features, they may be too complex or costly for small to medium-sized businesses. Additionally, they often require extensive training and may be overkill for businesses that only need basic inventory tracking.
* **How This Project Addresses Those Gaps**:  
  The Inventory Management System addresses these gaps by offering a simple, user-friendly interface with only the most essential features for managing product stock. This makes it ideal for small to medium-sized businesses that need a straightforward, cost-effective solution.

### **System Requirements**

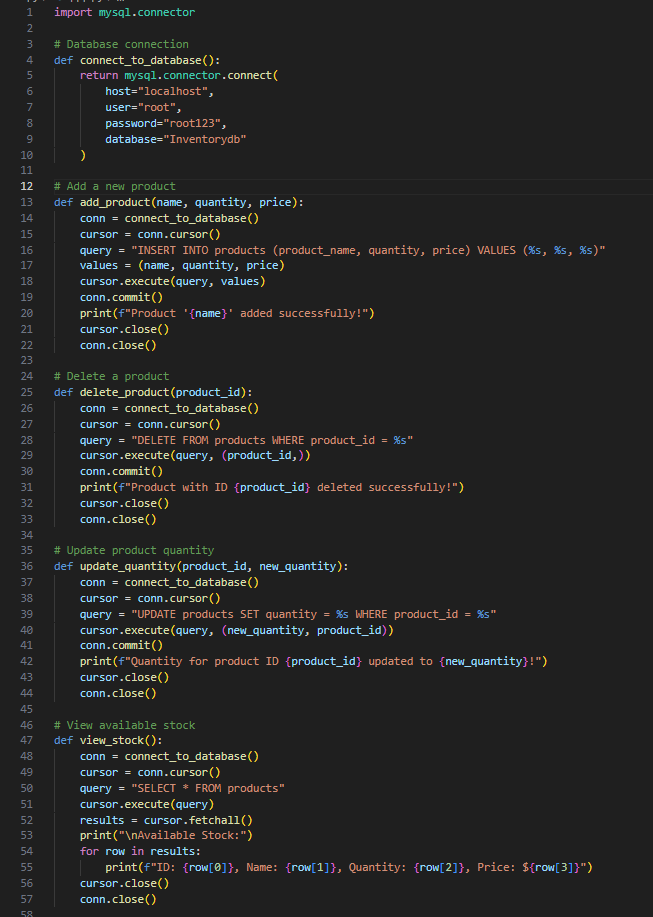
* **Hardware Requirements**:
  + Processor: Any modern processor (e.g., Intel Core i3 or higher)
  + RAM: Minimum of 4 GB
  + Storage: At least 500 MB of free space
  + Database: MySQL server installed on a local machine or remote server
* **Software Requirements**:
  + Programming Language: Python (version 3.6 or higher)
  + Database Management System: MySQL (version 5.7 or higher)
  + MySQL Connector for Python: For connecting Python to the MySQL database

**Implementation Details**

* **Technologies and Frameworks Used**:
  1. Python: Used for implementing the core logic of the application.
  2. MySQL: Used as the database to store and retrieve product data.
  3. MySQL Connector for Python: A Python library used to connect Python code to MySQL database.
* **Modules and Their Functionalities**:
  1. connect\_to\_database(): Establishes a connection to the MySQL database.
  2. add\_product(name, quantity, price): Adds a new product to the inventory.
  3. delete\_product(product\_id): Deletes a product based on its ID.
  4. update\_quantity(product\_id, new\_quantity): Updates the quantity of a specific product.
  5. view\_stock(): Displays all available products with their quantity and price.

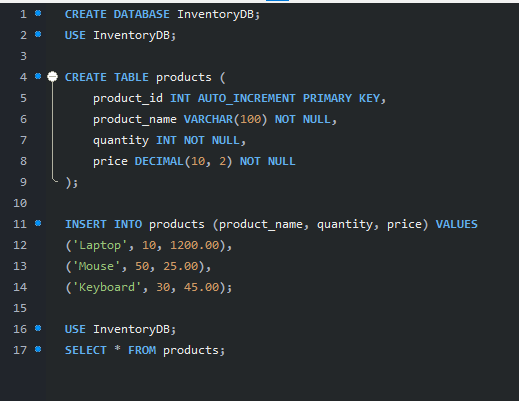
### **Python Code :**

Below is an example of the add\_product function:





### **SQL Code for Database Creation**



### **ER Diagram (Entity-Relationship Diagram)**

An **Entity-Relationship Diagram (ERD)** represents how different entities (or tables) in the system relate to each other. For your project, we have a **products table**.

Here’s a simple breakdown of the **Products Table**:

#### **Entities:**

* **Product**: Represents a product in the inventory.

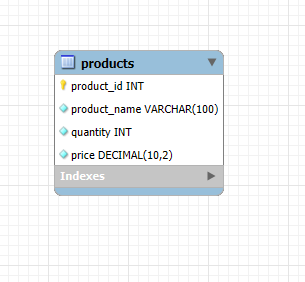
#### **Attributes:**

* **product\_id** (Primary Key): A unique identifier for each product.
* **product\_name**: The name of the product.
* **quantity**: The amount of stock available.
* **price**: The price of each product.

#### **Relationships:**

In this case, the relationship is simple: The **products** table stands alone with no complex relationships to other entities in the system.

Your **ERD** will have a single entity Product with attributes (columns), which would look like this:



### **Testing**

**Test Cases and Their Results**

1. **Test Case 1: Add Product**
   * **Input**: Product name: 'Laptop', Quantity: 10, Price: 1200.00
   * **Expected Result**: The product 'Laptop' is added to the inventory with the specified quantity and price.
   * **Actual Result**: Product added successfully.
2. **Test Case 2: Update Product Quantity**
   * **Input**: Product ID: 1, New Quantity: 15
   * **Expected Result**: The quantity of the product with ID 1 should be updated to 15.
   * **Actual Result**: Quantity updated successfully.

**Bugs and Their Fixes**

* **Bug 1**: In the initial version, the delete\_product function didn't handle invalid product IDs properly. This was fixed by adding an error check to ensure that the product ID exists before attempting deletion.

### **Results and Discussion**

* **Key Findings**:
  + The system is able to add, delete, and update product details accurately.
  + Real-time stock viewing works smoothly, providing immediate feedback to users.
* **Performance Analysis**:  
  The system performs well with minimal delay during database interactions, making it efficient for small to medium-sized businesses.
* **User Feedback**:  
  Users found the system intuitive and easy to use, appreciating the simplicity of the interface.

### **Conclusion**

* **Summary of the Project**:  
  The Inventory Management System provides a simple yet effective solution for managing inventory, allowing businesses to keep track of products, update quantities, and view current stock levels in real-time.
* **Challenges Faced**:
  + Initial database setup issues and handling connection timeouts.
  + Ensuring compatibility between Python and MySQL versions.
* **Future Enhancements**:
  + Implementing a graphical user interface (GUI) for a more user-friendly experience.
  + Adding additional features such as low-stock alerts and reporting functionalities.