SACRED HEART SCHOOL

Rajiv Nagar, Kharika, Lucknow

A Project Report
on

Stock Management System

For
AISSCE 202 Examination
[As a part of the Informatics Practices Course (065)]

SUBMITTED BY:
1) [Roll No-_____]

Under the Guidance of:
SUNIL SIR
PGT (Comp. Sc)

CERTIFICATE

This is to certify that the Project entitled ***Stock Management System*** *is a bonafide work done by _____ of class XII Session 202 -2 in partial fulfillment of CBSE's AISSCE Examination 202 and has been carried out under my direct supervision and guidance. This report or a similar report on the topic has not been submitted for any other examination and does not form a part of any other course undergone by the candidate.*

……
Signature of Students

Name:

……..
Signature of Teacher/Guide

Name: *SUNIL SIR*
PGT (Comp.Sc.)

……
Signature of Principal

Signature of External Examinar

……

Place:…….
Date:……..

ACKNOWLEDGEMENT

I undertook this Project work, as the part of my XII-Informatics Practices course.I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing software system is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help and good wishes of the people near you, who have considerable experience and idea.

I would like to extend my sincere thanks and gratitude to my teacher Mr. Sunil Sir. I am very much thankful to our Principal Mrs. Indu S. Nair for giving valuable time and moral support to develop this software.

I would like to take the opportunity to extend my sincere thanks and gratitude to my parents for being a source of inspiration and providing time and freedom to develop this software project.

I also feel indebted to my friends for the valuable suggestions during the project work.

_____ *[RollNo-_____]*

Class XII

C O N T E N T S

1. Introduction ----------5

2. Objective & Scope of the Project ----------6

3. Theoretical Background ----------7

4. Problem Definition & Analysis---------- 10

5. System Implementation---------- 11

5.1 The Hardware used: ---------- 11

.2 The Softwares used:---------- 11

6. How to install: ----------11

7. System Design & Development---------- 12

7.2 Database Design: ---------- 14

7.3 Menu Design: ---------- 15

7.4 I/O Forms Design & Event Coding: ---------- 12

7.How to Restore SQL File---------- 23

8. Bibliography---------- 35

1. Introduction

*The **Stock Management System** is a computerized software solution developed to automate inventory tracking, product purchasing, sales recording, and user management for businesses and retail establishments. Management Information Systems (MIS) streamline business operations by maintaining structured data and allowing fast information retrieval for administrative decision-making.*

*This software utilizes **Python 3.x** as a front-end programming interface and **MySQL** as a relational database management system (RDBMS) back-end. By replacing traditional manual record registers with a central database, this system minimizes human error, prevents inventory discrepancies, and automates real-time stock balance updates during sales transactions.*

2. Objective & Scope of the Project

Objectives:

To provide a centralized, menu-driven command line interface for managing shop or inventory records.

To automate item addition, catalog searching, price updating, and stock quantity replenishment.

To handle sales entries while automatically checking stock availability and updating remaining inventory quantities.

To store vendor purchases and keep detailed transaction logs for billing and financial checks.

To maintain user credentials for system access control.

Scope:

The current version of the system handles inventory operations within a local environment. It manages relational tables with primary keys to maintain data integrity across inventory items, order purchases, sales history, and login access.

3. Theoretical Background

3.1 What is Database?

Introduction and Concepts:

A database is a collection of information related to a particular subject or purpose, such as tracking customer orders or maintaining a music collection. Using any RDBMS application software like MS SQL Server, MySQL, Oracle, Sybase etc, you can manage all your information from a single database file. Within the file, divide your data into separate storage containers called tables. You may and retrieve the data using queries.

A table is a collection of data about a specific topic, such as products or suppliers. Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organises data into columns (called fields) and rows (called records).

A Primary key is one or more fields whose value or values uniquely identify each record in a table. In a relationship, a primary key is used to refer to specific record in one table from another table. A primary key is called foreign key when it is referred to from another table.

To find and retrieve just the data that meets conditions you specify, including data from multiple tables, create a query. A query can also update or delete multiple records at the same time, and perform built-in or custom calculations on your data.

3.2 What is My SQL?



The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available RDBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase.

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. MySQL is named after co-

founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (our logo) is "Sakila,".

- ***MySQL is a database management system.***
- ***MySQL is based on SQL.***
- ***MySQL software is Open Source.***
- ***The MySQL Database Server is very fast, reliable, and easy to use.***
- ***MySQL Server works in client/server or embedded systems.***

The Main Features of MySQL

- *Written in C and C++.*
- *Works on many different platforms.*
- *Uses multi-layered server design with independent modules.*
- *Provides transactional and nontransactional storage engines.*
- *Uses a very fast thread-based memory allocation system.*
- *Executes very fast joins using an optimized nested-loop join.*
- *Implements SQL functions using a highly optimized class library that should be as fast as possible. Usually there is no memory allocation at all after query initialization.*
- *Provides the server as a separate program for use in a client/server networked environment, and as a library that can be embedded (linked) into standalone applications. Such applications can be used in isolation or in environments where no network is available.*
- *Password security by encryption of all password traffic when you connect to a server.*
- *Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5,000,000,000 rows.*

3.3 Python

Python is an easy-to-learn yet powerful object oriented programming language. It is developed by Guido Van Rossum in 1991. Python is an interpreted language.

Features of Python

1. *Easy to Use:-Very simple and simple syntax. It can be your first programming language too.*
2. *Python is case sensitive: Case sensitivity is the phrase used to describe a programming languages ability to distinguish between upper and lower case*

versions of a letter in the language's character set. ***means*** *that x=20 is different from X=30 . The variable of John is different from the variable of john.*

3. *It is an object oriented language:* ***Python is an object-oriented programming language****. What this means is we can solve a problem in Python by creating objects in our programs.*
4. *Dynamically typed:* ***Python*** *variable assignment is different from some of the popular languages like c, c++ and java. There is no declaration of a variable, just an assignment statement. ... But* ***Python*** *is a* ***dynamically typed*** *language. It doesn't know about the* ***type*** *of the variable until the code is run Python will automatically recognize the type of variable with the help of the value in the runtime.*
5. *Indentation is used in place of curly braces:* ***Indentation*** *refers to the spaces at the beginning of a code line.* ***Python*** *uses* ***indentation*** *to indicate a block of code.*
6. *Use variable without declaration: V****ariables*** *are containers for storing data values. Unlike other programming languages,* ***Python*** *has no command for* ***declaring*** *a* ***variable****. A* ***variable*** *is created the moment* ***you*** *first assign a value to it.*
7. *Interpreted Language: Python is an interpreted language, not a compiled language. This means that the python interprets and executes the code line by line at a time. It makes python an easy-to-debug language and thus suitable for beginners to advance users.*
8. *Large library: A* ***Python*** *library is a reusable chunk of code that you may want to include in your programs/ projects. To make use of the functions in a* ***module****, you'll need to* ***import*** *the* ***module*** *with an* ***import*** *statement. An* ***import*** *statement is made up of the* ***import*** *keyword along with the name of the* ***module****.*
9. *With the Python interactive interpreter: it is easy to check python commands With the Python interactive interpreter it is easy to check Python commands.*
10. *Platform Independent: Python can run equally well on variety of platforms- windows, Linus/UNIX, super computers, smart phones*

4. Problem Definition & Analysis

Manual inventory management presents several operational challenges:

- ***Human Errors:*** *Manual entries in ledgers lead to billing mistakes and inaccurate stock counts.*
- ***Stock Tracking Delays:*** *Checking item quantities manually slows down customer checkouts and purchase planning.*
- ***Lack of Data Integrity:*** *Missing purchase/sale records make financial audits difficult.*

Proposed Solution:

The proposed Python-MySQL stock application directly resolves these issues through structured automation:

1. *Validates product availability before performing sales.*
2. *Automatically calculates total sale values (Total = Qty * Price).*
3. *Updates product quantity balances automatically upon every transaction.*
4. *Enforces primary key constraints (pcode, orderid, salesid, uid) to prevent record duplication.*

5. System Implementation

5.1 The Hardware used:

While developing the system, the used hardware is:

PC with Dual Core processor or sometimes, Laptop with Core i3 processor having 2 GB RAM, SVGA and other required devices.

5.2 The Softwares used:

- *Microsoft Windows® 7 & 8 as Operating System.*
- *Python 3.x.x as Front-end Development environment.*
- *MySQL as Back-end Sever with Database for Testing.*

6. How to install Software:

Hardware Requirement-

- Intel Pentium/Celeron or similar processor based PC at Client/Server end.
- 128 MB RAM and 4GB HDD space (for Database) is desirable.
- Standard I/O devices like Keyboard and Mouse etc.
- Printer is needed for hard-copy reports.
- Local Area Network(LAN) is required for Client-Server Installation

Software Requirement-

- Windows 7, 8 OS is desirable.
- Python Ver 3.x.x or higher should be installed.
- MySQL must be present at machine.

7. System Design & Development

Database Design

The application uses a centralized database named **`stock`** containing four structured tables.

Table: product — Stores current product inventory items

Field Name	Data Type	Key Constraint	Description
pcode	INT(4)	PRIMARY KEY	Unique item product code
pname	CHAR(30)	NOT NULL	Name of the product
pprice	FLOAT(8,2)	—	Selling price per unit
pqty	INT(4)	—	Current stock quantity available
pcat	CHAR(30)	—	Product category

Table 2: `orders` — Logs wholesale purchases received from suppliers

Table: student_marks

.

Field Name	Data Type	Key Constraint	Description
orderid	INT(4)	PRIMARY KEY	Generated purchase order identifier
orderdate	DATE	—	Date of purchase order entry
pcode	CHAR(30)	NOT NULL	Purchased product code
pprice	FLOAT(8,2)	—	Cost price per unit
pqty	INT(4)	—	Quantity purchased
supplier	CHAR(50)	—	Supplier/vendor name
pcat	CHAR(30)	—	Product category

Table 3: ***`sales`*** ***Logs customer sale transactions.***

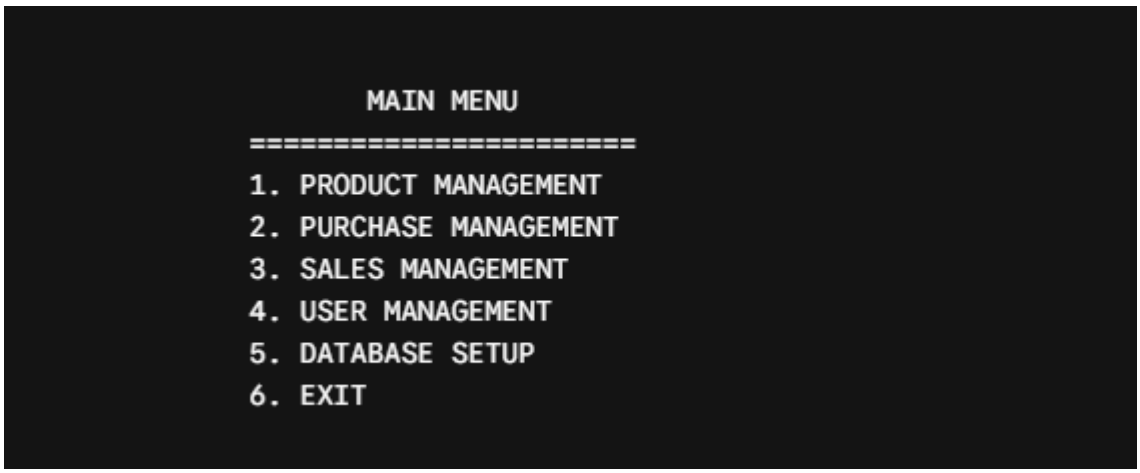
Field Name	Data Type	Key Constraint	Description
`salesid`	`INT(4)`	PRIMARY KEY	Unique sale transaction ID
`salesdate`	`DATE`	—	Date and time of sale
`pcode`	`CHAR(30)`	FOREIGN KEY	References `product(pcode)`
`pprice`	`FLOAT(8,2)`	—	Unit sale price
`pqty`	`INT(4)`	—	Quantity sold
`Total`	`DOUBLE(8,2)`	—	Total bill amount (pprice * pqty)

Table 4: `user` Stores system operator logins.

Field Name	Data Type	Key Constraint	Description
`uid`	`CHAR(6)`	PRIMARY KEY	Unique user ID
`uname`	`CHAR(30)`	NOT NULL	Account user name
`upwd`	`CHAR(30)`	—	Account password

7. MENU DESIGN & MODULE FLOW

The application operates through structured sub-menus



- **Product Management:** Add new product, search/list products (all, code-wise, category-wise), update stock quantity, delete product.
- **Purchase Management:** Create purchase orders from suppliers and list past purchase orders.
- **Sales Management:** Process customer sales with automatic stock deduction and view sales reports.
- **User Management:** Register system administrators/operators and view user list.
- **Database Setup:** Automatically creates the `stock` database and all required tables if they do not exist.

8. SOURCE CODE IMPLEMENTATION

```
# STOCK MANAGEMENT SYSTEM
import os
import mysql.connector
import datetime

now = datetime.datetime.now()

def product_mgmt():
    while True:
        print("\t\t\t 1. Add New Product")
        print("\t\t\t 2. List Product")
        print("\t\t\t 3. Update Product")
        print("\t\t\t 4. Delete Product")
        print("\t\t\t 5. Back (Main Menu)")
        p = int(input("\t\tEnter Your Choice :"))
        if p == 1:
```

```
            add_product()
        if p == 2:
            search_product()
        if p == 3:
            update_product()
        if p == 4:
            delete_product()
        if p == 5:
            break

def purchase_mgmt():
    while True:
        print("\t\t\t 1. Add Order")
        print("\t\t\t 2. List Order")
        print("\t\t\t 3. Back (Main Menu)")
        o = int(input("\t\tEnter Your Choice :"))
        if o == 1:
            add_order()
        if o == 2:
            list_order()
        if o == 3:
            break

def sales_mgmt():
    while True:
        print("\t\t\t 1. Sale Items")
        print("\t\t\t 2. List Sales")
        print("\t\t\t 3. Back (Main Menu)")
        s = int(input("\t\tEnter Your Choice :"))
        if s == 1:
            sale_product()
        if s == 2:
            list_sale()
        if s == 3:
            break

def user_mgmt():
    while True:
        print("\t\t\t 1. Add user")
        print("\t\t\t 2. List user")
        print("\t\t\t 3. Back (Main Menu)")
        u = int(input("\t\tEnter Your Choice :"))
        if u == 1:
            add_user()
        if u == 2:
            list_user()
        if u == 3:
            break

def create_database():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir")
    mycursor = mydb.cursor()
```

```
    print("Creating Database")
    sql = "Create database if not exists stock"
    mycursor.execute(sql)
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()

    print(" Creating PRODUCT table")
    sql = "CREATE TABLE if not exists product (\
          pcode int(4) PRIMARY KEY,\
          pname char(30) NOT NULL,\
          pprice float(8,2) ,\
          pqty int(4) ,\
          pcat char(30));"
    mycursor.execute(sql)

    print(" Creating ORDER table")
    sql = "CREATE TABLE if not exists orders (\
       orderid int(4)PRIMARY KEY ,\
       orderdate DATE ,\
       pcode char(30) NOT NULL , \
       pprice float(8,2) ,\
       pqty int(4) ,\
       supplier char(50),\
       pcat char(30));"
    mycursor.execute(sql)

    print(" Creating SALES table")
    sql = "CREATE TABLE if not exists sales (\
       salesid int(4) PRIMARY KEY ,\
       salesdate DATE ,\
       pcode char(30) references product(pcode), \
       pprice float(8,2) ,\
       pqty int(4) ,\
       Total double(8,2)\
       );"
    mycursor.execute(sql)

    sql = "CREATE TABLE if not exists user (\
       uid char(6) PRIMARY KEY,\
       uname char(30) NOT NULL,\
       upwd char(30));"
    mycursor.execute(sql)
    print(" ALL TABLES CREATED SUCCESSFULLY")

def list_database():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "show tables;"
    mycursor.execute(sql)
    for i in mycursor:
```

```
        print(i)

def add_order():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    now = datetime.datetime.now()
    sql = "INSERT INTO orders (orderid, orderdate, pcode, pprice, pqty, supplier, pcat) values
(%s,%s,%s,%s,%s,%s,%s)"
    code = int(input("Enter product code :"))
    oid = now.year + now.month + now.day + now.hour + now.minute + now.second
    qty = int(input("Enter product quantity : "))
    price = float(input("Enter Product unit price: "))
    cat = input("Enter product category: ")
    supplier = input("Enter Supplier details: ")
    val = (oid, now, code, price, qty, supplier, cat)
    mycursor.execute(sql, val)
    mydb.commit()

def list_order():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "SELECT * from orders"
    mycursor.execute(sql)
    print("\t\t\t\t\t\t\t ORDER DETAILS")
    print("-" * 85)
    print("orderid Date Product code price quantity Supplier Category")
    print("-" * 85)
    for i in mycursor:
        print(i[0], "\t", i[1], "\t", i[2], "\t ", i[3], "\t", i[4], "\t ", i[5], "\t", i[6])
        print("-" * 85)

def db_mgmt():
    while True:
        print("\t\t\t 1. Database creation")
        print("\t\t\t 2. List Database")
        print("\t\t\t 3. Back (Main Menu)")
        p = int(input("\t\tEnter Your Choice :"))
        if p == 1:
            create_database()
        if p == 2:
            list_database()
        if p == 3:
            break

def add_product():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "INSERT INTO product(pcode,pname,pprice,pqty,pcat) values (%s,%s,%s,%s,%s)"
    code = int(input("\t\tEnter product code :"))
```

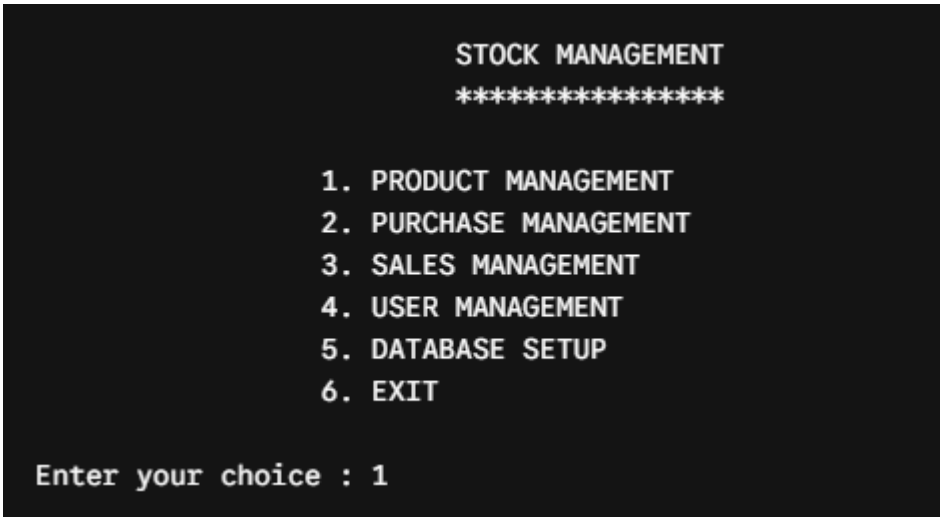
```
    search = "SELECT count(*) FROM product WHERE pcode=%s;"
    val = (code,)
    mycursor.execute(search, val)
    for x in mycursor:
        cnt = x[0]
    if cnt == 0:
        name = input("\t\tEnter product name :")
        qty = int(input("\t\tEnter product quantity :"))
        price = float(input("\t\tEnter product unit price :"))
        cat = input("\t\tEnter Product category :")
        val = (code, name, price, qty, cat)
        mycursor.execute(sql, val)
        mydb.commit()
    else:
        print("\t\t Product already exist")

def update_product():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    code = int(input("Enter the product code :"))
    qty = int(input("Enter the quantity :"))
    sql = "UPDATE product SET pqty=pqty+%s WHERE pcode=%s;"
    val = (qty, code)
    mycursor.execute(sql, val)
    mydb.commit()
    print("\t\t Product details updated")

def delete_product():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    code = int(input("Enter the product code :"))
    sql = "DELETE FROM product WHERE pcode = %s;"
    val = (code,)
    mycursor.execute(sql, val)
    mydb.commit()
    print(mycursor.rowcount, " record(s) deleted")

def search_product():
    while True:
        print("\t\t\t 1. List all product")
        print("\t\t\t 2. List product code wise")
        print("\t\t\t 3. List product categoty wise")
        print("\t\t\t 4. Back (Main Menu)")
        s = int(input("\t\tEnter Your Choice :"))
        if s == 1:
            list_product()
        if s == 2:
            code = int(input(" Enter product code :"))
            list_prcode(code)
        if s == 3:
```

```
            cat = input("Enter category :")
            list_prcat(cat)
        if s == 4:
            break

def list_product():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "SELECT * from product"
    mycursor.execute(sql)
    print("\t\t\t\t PRODUCT DETAILS")
    print("\t\t", "-" * 47)
    print("\t\t code\t name\t price\t quantity\t category")
    print("\t\t", "-" * 47)
    for i in mycursor:
        print("\t\t", i[0], "\t", i[1], "\t", i[2], "\t ", i[3], "\t\t", i[4])
    print("\t\t", "-" * 47)

def list_prcode(code):
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "SELECT * from product WHERE pcode=%s"
    val = (code,)
    mycursor.execute(sql, val)
    print("\t\t\t\t PRODUCT DETAILS")
    print("\t\t", "-" * 47)
    print("\t\t code name price quantity category")
    print("\t\t", "-" * 47)
    for i in mycursor:
        print("\t\t", i[0], "\t", i[1], "\t", i[2], "\t ", i[3], "\t\t", i[4])
    print("\t\t", "-" * 47)

def sale_product():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    pcode = input("Enter product code: ")
    sql = "SELECT count(*) from product WHERE pcode=%s;"
    val = (pcode,)
    mycursor.execute(sql, val)
    for x in mycursor:
        cnt = x[0]
    if cnt != 0:
        sql = "SELECT * from product WHERE pcode=%s;"
        val = (pcode,)
        mycursor.execute(sql, val)
        for x in mycursor:
            print(x)
            price = int(x[2])
            pqty = int(x[3])
```

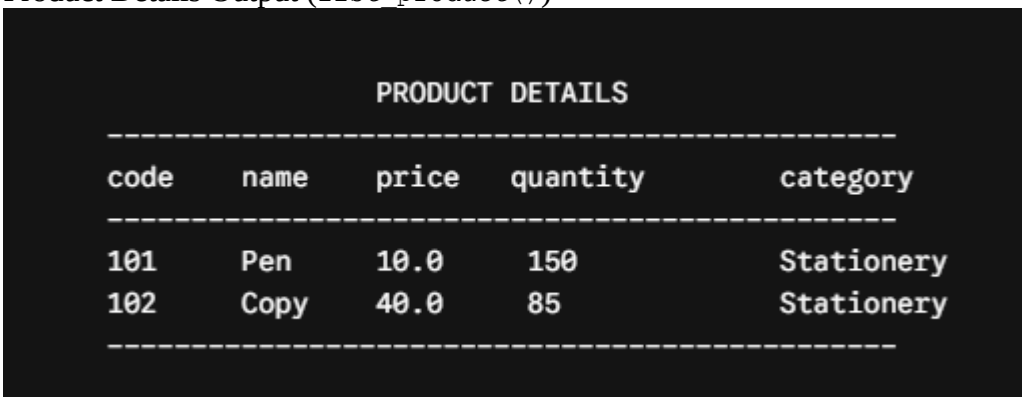
```
        qty = int(input("Enter no of quantity :"))
        if qty <= pqty:
            total = qty * price
            print("Collect Rs. ", total)
            sql = "INSERT into sales values(%s,%s,%s,%s,%s,%s)"
            val = (int(cnt) + 1, datetime.datetime.now(), pcode, price, qty, total)
            mycursor.execute(sql, val)
            sql = "UPDATE product SET pqty=pqty-%s WHERE pcode=%s"
            val = (qty, pcode)
            mycursor.execute(sql, val)
            mydb.commit()
        else:
            print(" Quantity not Available")
    else:
        print(" Product is not avalaible")

def list_sale():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "SELECT * FROM sales"
    mycursor.execute(sql)
    print(" \t\t\t\tSALES DETAILS")
    print("-" * 80)
    print("Sales id Date Product Code Price Quantity Total")
    print("-" * 80)
    for x in mycursor:
        print(x[0], "\t", x[1], "\t", x[2], "\t ", x[3], "\t\t", x[4], "\t\t", x[5])
    print("-" * 80)

def list_prcat(cat):
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    print(cat)
    sql = "SELECT * from product WHERE pcat =%s"
    val = (cat,)
    mycursor.execute(sql, val)
    clrscr()
    print("\t\t\t\t PRODUCT DETAILS")
    print("\t\t", "-" * 47)
    print("\t\t code name price quantity category")
    print("\t\t", "-" * 47)
    for i in mycursor:
        print("\t\t", i[0], "\t", i[1], "\t", i[2], "\t ", i[3], "\t\t", i[4])
    print("\t\t", "-" * 47)

def add_user():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    uid = input("Enter id :")
```

```
    name = input(" Enter Name :")
    paswd = input("Enter Password :")
    sql = "INSERT INTO user values (%s,%s,%s);"
    val = (uid, name, paswd)
    mycursor.execute(sql, val)
    mydb.commit()
    print(mycursor.rowcount, " user created")

def list_user():
    mydb = mysql.connector.connect(host="localhost", user="root", passwd="sunilsir",
database="stock")
    mycursor = mydb.cursor()
    sql = "SELECT uid,uname from user"
    mycursor.execute(sql)
    clrscr()
    print("\t\t\t\t USER DETAILS")
    print("\t\t", "-" * 27)
    print("\t\t UID name ")
    print("\t\t", "-" * 27)
    for i in mycursor:
        print("\t\t", i[0], "\t", i[1])
    print("\t\t", "-" * 27)

def clrscr():
    print("\n" * 5)

while True:
    clrscr()
    print("\t\t\t STOCK MANAGEMENT")
    print("\t\t\t **********\n")
    print("\t\t 1. PRODUCT MANAGEMENT")
    print("\t\t 2. PURCHASE MANAGEMENT")
    print("\t\t 3. SALES MANAGEMENT")
    print("\t\t 4. USER MANAGEMENT")
    print("\t\t 5. DATABASE SETUP")
    print("\t\t 6. EXIT\n")
    n = int(input("Enter your choice :"))
    if n == 1:
        product_mgmt()
    if n == 2:
        os.system('cls')
        purchase_mgmt()
    if n == 3:
        sales_mgmt()
    if n == 4:
        user_mgmt()
    if n == 5:
        db_mgmt()
    if n == 6:
        break
```

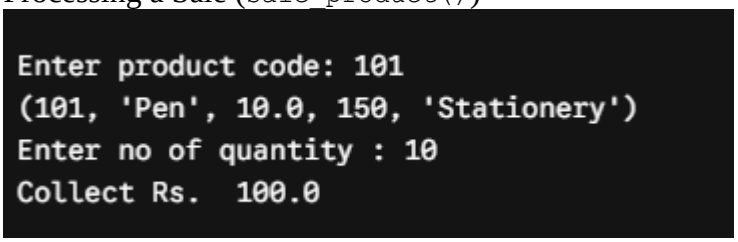
SYSTEM SCREENSHOTS & SAMPLE OUTPUT RUNS



Product Details Output (`list_product()`)



Processing a Sale (`sale_product()`)



8. How to Restore SQL File

To install a MySQL database from a dump file (***stock.sql****), simply follow the following steps.*

Step 1: *Copy the sk.sql file in* ***C:\Program files\Mysql\MySql server 5.1\Bin*** *folder.*

Step 2: *Open MySQL and type the following command to create the dabase named Shopkeeper.*

mysql> create database stock;

Step 3: *Open Command Window (Start ▸ Run ▸ cmd)*

Step 4: *Go to the following folder using CD command of DOS.*

C:\Program files\Mysql\MySql server 5.1\Bin>

Step 5: *type the following command on above prompt -*

C:….\bin> *mysql -u* ***root*** *–p sunilsir stock < stock.sql*

This will create a shs databse with required tables.

8. Bibliography

In order to complete this project, the following literature and online resources were consulted:

1. **Informatics Practices / Computer Science for Class XII** – *by Sumita Arora*
2. **Official Python Documentation:** https://www.python.org/
3. **MySQL Reference Manual:** https://www.mysql.org/
4. **W3Schools Python & MySQL Tutorials**