Inventory Management

&

Customer Billing System

Student Name: PRISHA GABBAD
Reg. No.: 25BCCY10139

Subject: CSE1021 INTRODUCTION TO PROBLEM SOLVING

College: Vellore Institute of Technology BHOPAL

Year: 2025

Submitted To: Dr. Lokesh Malviya

Slot: C14+E11+E12

Introduction

This project is a Python-based Inventory Management and Customer Billing System developed using MySQL as the backend database. The system automates stock handling, updates product quantities, generates customer bills, and ensures smooth retail operations. It eliminates manual bookkeeping and provides accurate, real-time record management.

Problem Statement

Many retail shops and small businesses still maintain stock manually, which causes errors in quantity tracking and billing. There is a need for a simple, fast, and efficient computerized system to: Maintain product stock, Update items after purchase, Generate correct bills, Reduce human error.

This project solves these problems through a structured, database-driven application.

Functional Requirements

The system performs the following functions:

Stock Module

1. Display all stock items
2. Add new stock items
3. Delete an existing item
4. Modify/update a stock item
5. Display details of a single item

Billing Module

1. Search stock from database
2. Verify available quantity
3. Generate bill for customer
4. Update stock after billing
5. Save bill details into bill.csv

Non-Functional Requirements

1. **Usability**: User-friendly menu-driven interface via console.

2. **Performance**: Fetches and updates stock within milliseconds using MySQL.

3. **Reliability**: All operations are database-committed to ensure accuracy.

4. **Maintainability**: Code is modular—stock() and billing() modules separated.

5. **Security**: Database connection protected by MySQL credentials.

6. **Error Handling**: Handles invalid input, missing items, and insufficient stock.

System Architecture

Architecture Layers:

1. User Interface: Python CLI

2. Application Logic:

Stock maintenance module

Billing module

3. Database Layer: MySQL stock table

4. File Storage: bill.csv

Flow:

User → Python Program → MySQL Database → Updated Stock → Bill Generated

Design Diagrams

a. Use Case Diagram

Manage Stock

Add/Modify/Delete Item

Generate Bill

Update Stock

Save Bill

b. Workflow Diagram

Start

↓

Main Menu

↓

[1] Stock Module → Display/Add/Delete/Modify

↓

[2] Billing Module → Select Item → Check Qty → Calculate Bill → Update Stock → Save CSV

↓

Exit

c. Sequence Diagram (Billing Example)

User → System: Enter item_no

System → DB: fetch item

DB → System: return item

User → System: enter qty

System → System: calculate total

System → DB: update stock

System → CSV: write bill

System → User: display bill

d. Class/Component Diagram

Components:

get_db_connection()

stock()

billing()

MySQL stock table

bill.csv file

e. ER Diagram

Table: STOCK

item_no

item_name

quantity

unit_price

Design Decisions & Rationale

MySQL chosen for reliability and persistent storage.

CSV used for bills because it is portable and readable in Excel.

Modular functions used to improve maintainability.

Loop-based UI ensures repeated operations without restarting the program.

Implementation Details

Python used for main program.

mysql.connector used for DB interactions.

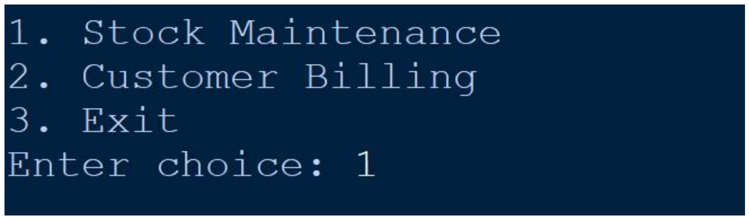
Menu-driven console interface.

Exception handling for invalid inputs.

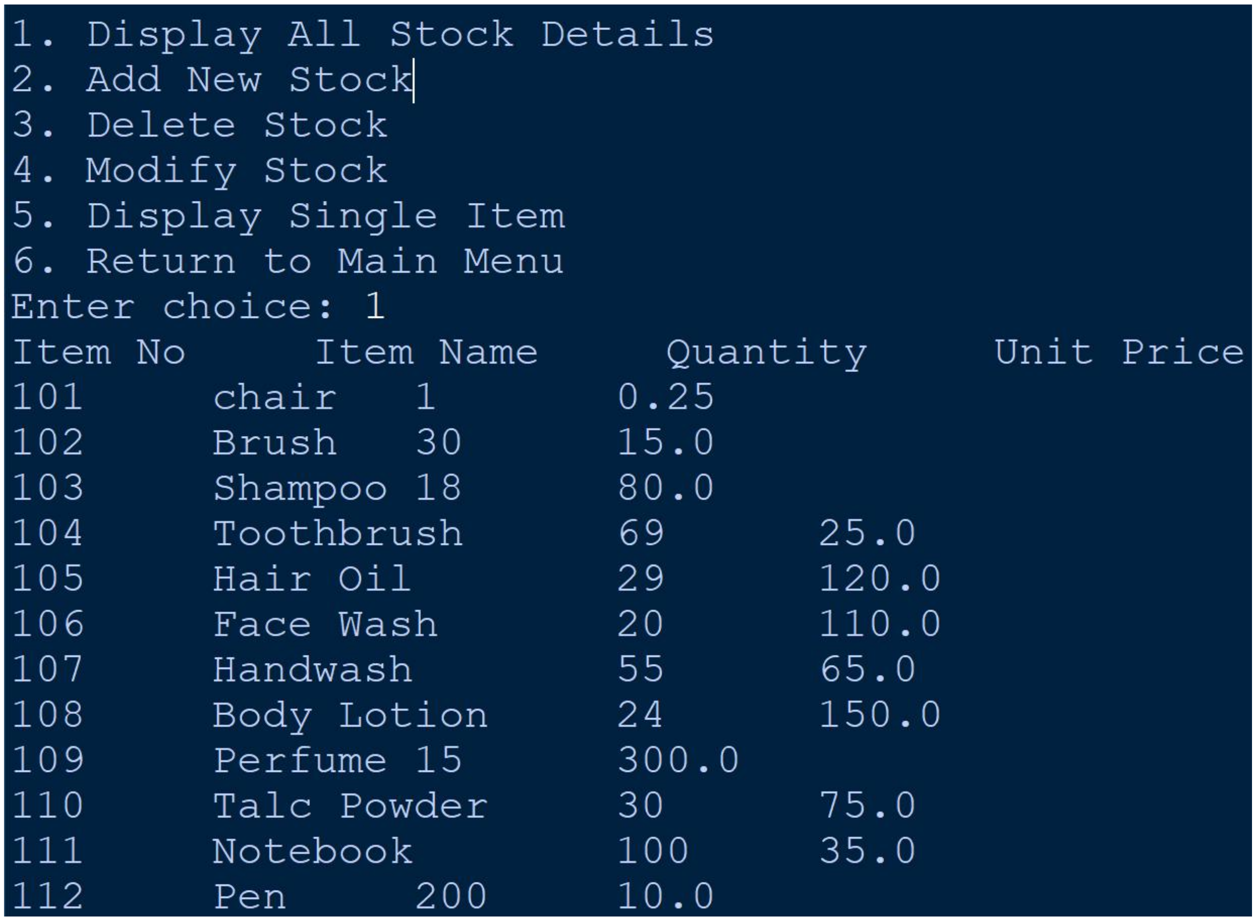
Auto-update of stock after billing.

bill.csv generated with multiple entries (customer_name, item_no, name, qty, unit_price, total).

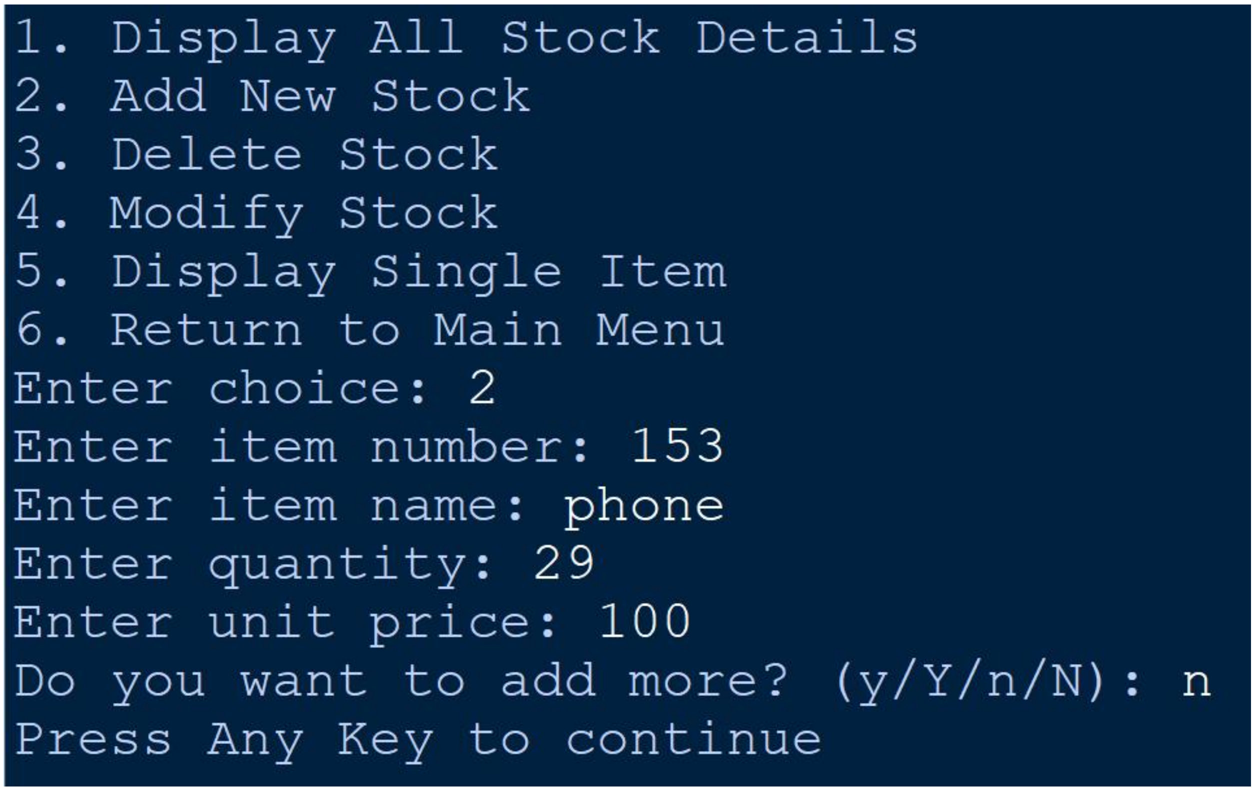
Results



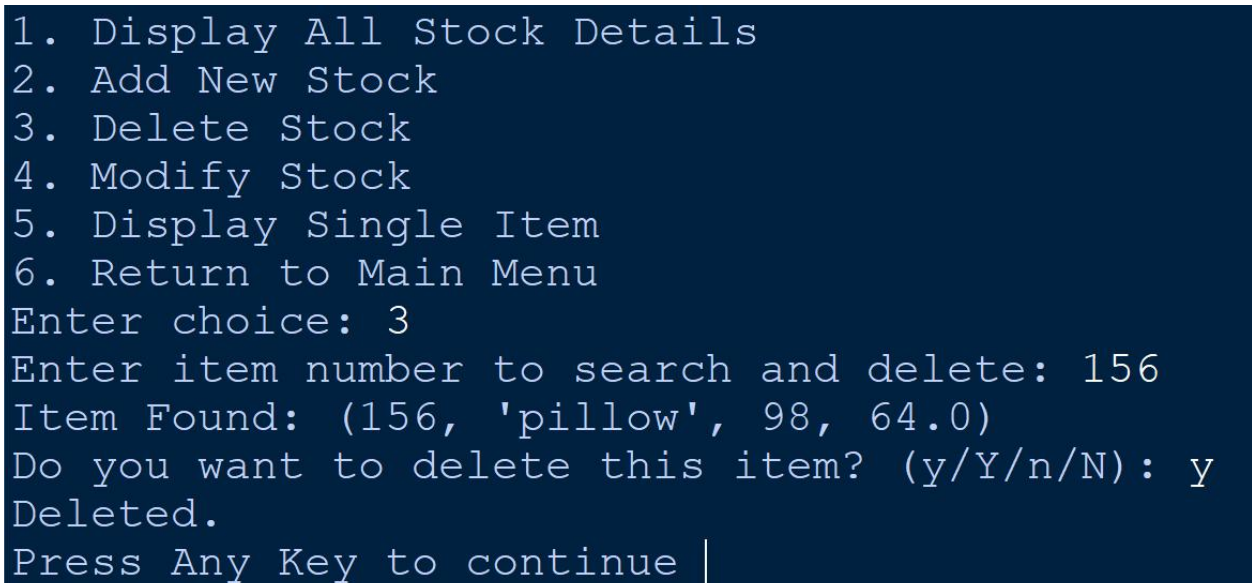
```
1. Stock Maintenance
2. Customer Billing
3. Exit
Enter choice: 1
```



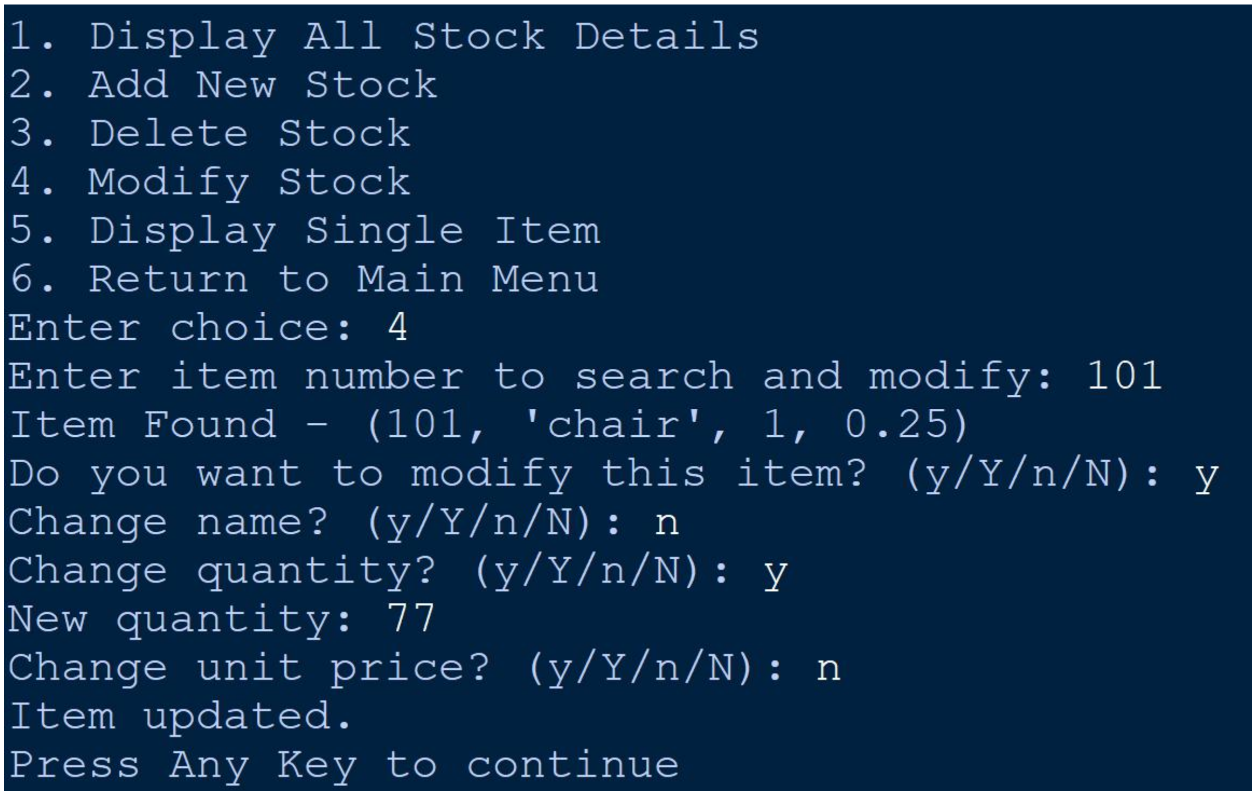
```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 1
Item No      Item Name     Quantity     Unit Price
101     chair   1       0.25
102     Brush   30      15.0
103     Shampoo 18      80.0
104     Toothbrush      69      25.0
105     Hair Oil        29      120.0
106     Face Wash       20      110.0
107     Handwash        55      65.0
108     Body Lotion     24      150.0
109     Perfume 15      300.0
110     Talc Powder     30      75.0
111     Notebook        100     35.0
112     Pen     200     10.0
```



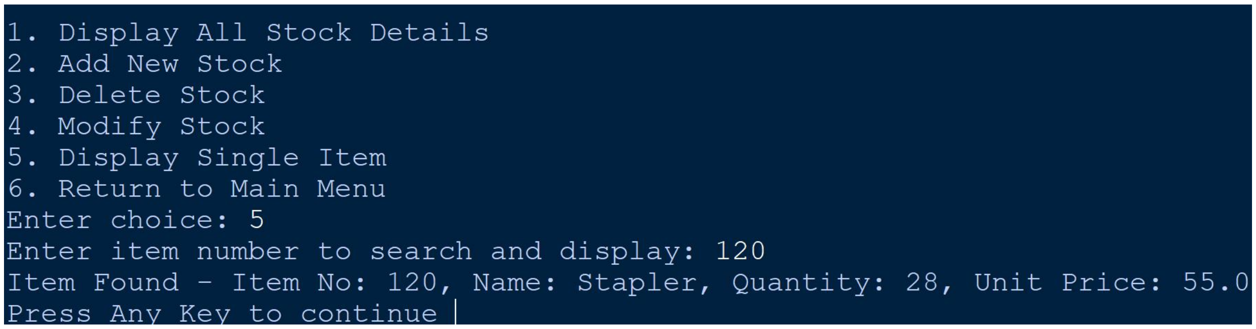
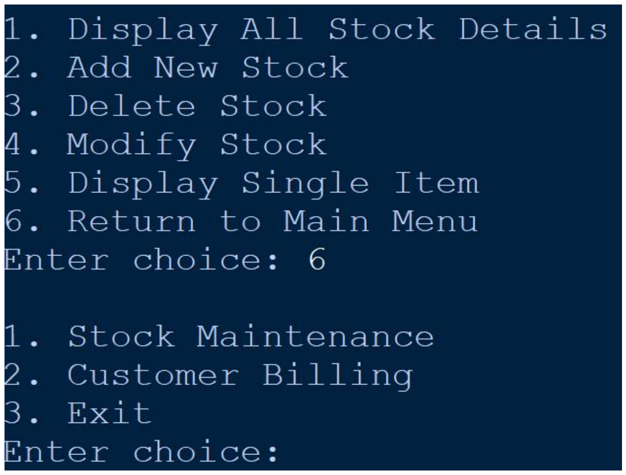
```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 2
Enter item number: 153
Enter item name: phone
Enter quantity: 29
Enter unit price: 100
Do you want to add more? (y/Y/n/N): n
Press Any Key to continue
```

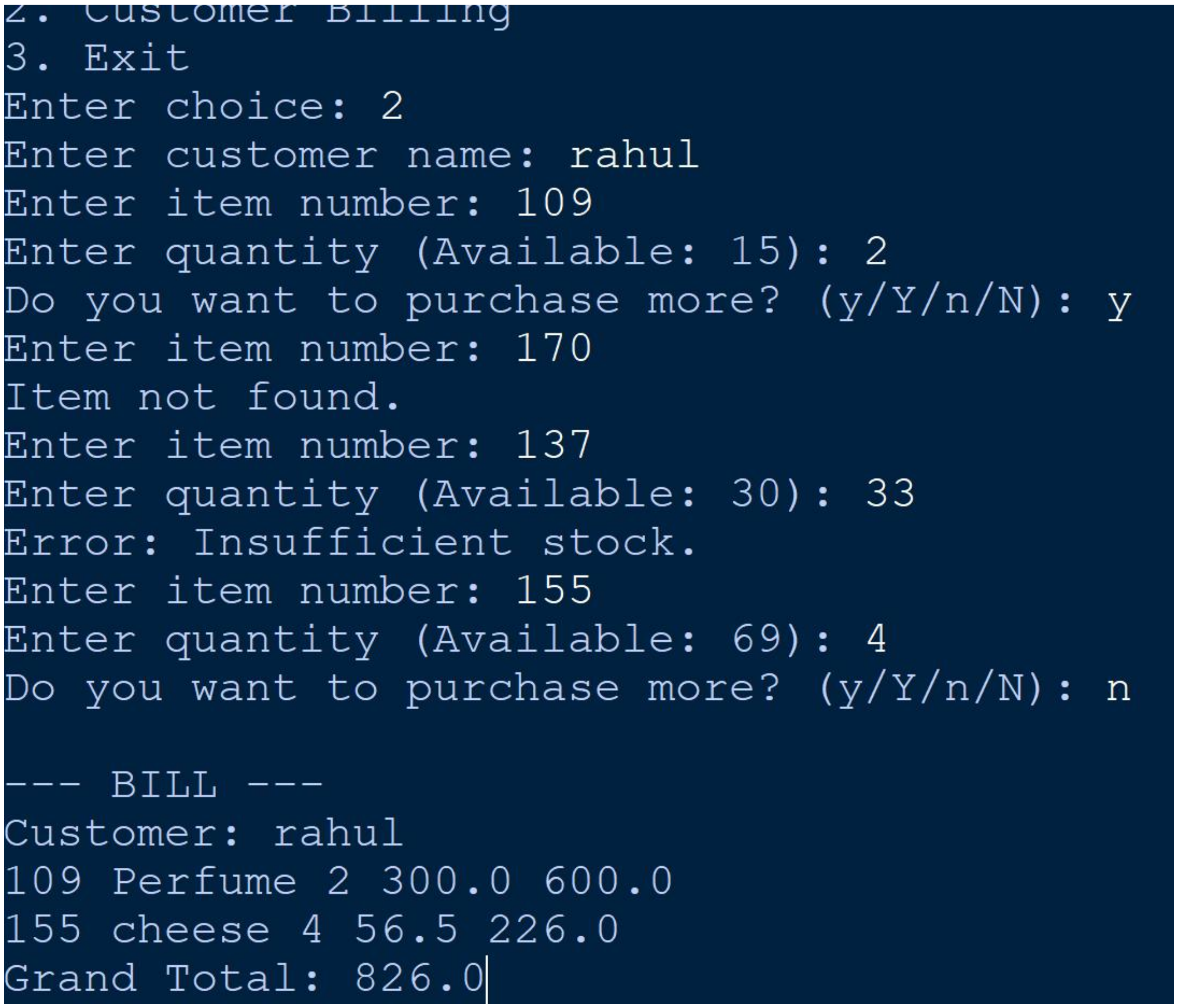


```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 3
Enter item number to search and delete: 156
Item Found: (156, 'pillow', 98, 64.0)
Do you want to delete this item? (y/Y/n/N): y
Deleted.
Press Any Key to continue
```

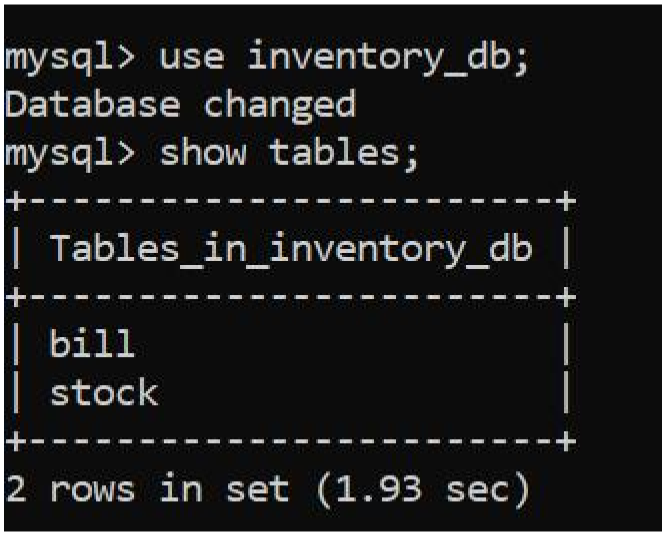


```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 4
Enter item number to search and modify: 101
Item Found - (101, 'chair', 1, 0.25)
Do you want to modify this item? (y/Y/n/N): y
Change name? (y/Y/n/N): n
Change quantity? (y/Y/n/N): y
New quantity: 77
Change unit price? (y/Y/n/N): n
Item updated.
Press Any Key to continue
```

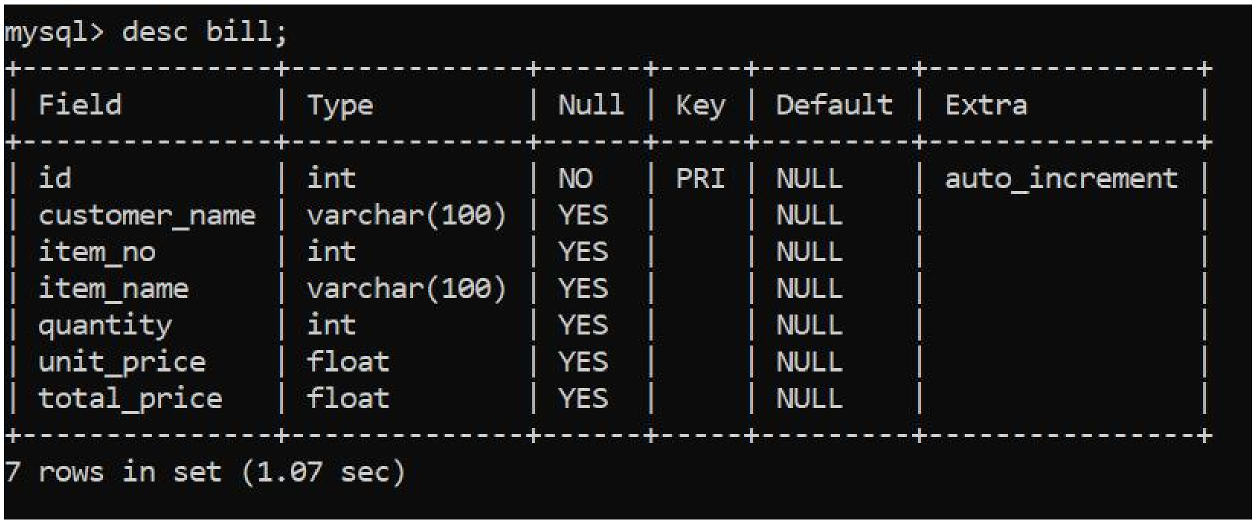

```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 5
Enter item number to search and display: 120
Item Found - Item No: 120, Name: Stapler, Quantity: 28, Unit Price: 55.0
Press Any Key to continue
```

```
1. Display All Stock Details
2. Add New Stock
3. Delete Stock
4. Modify Stock
5. Display Single Item
6. Return to Main Menu
Enter choice: 6

1. Stock Maintenance
2. Customer Billing
3. Exit
Enter choice:
```



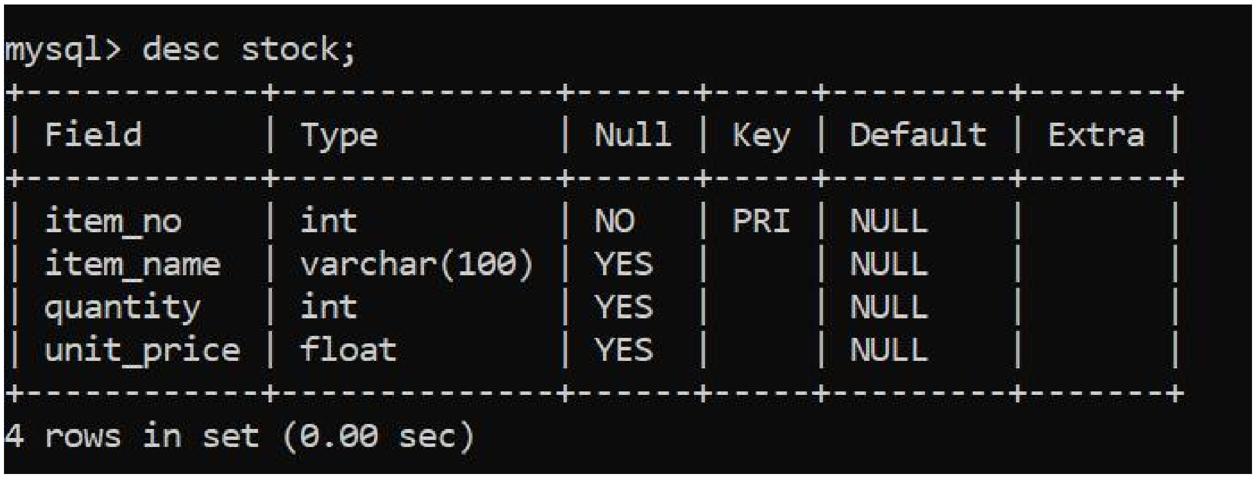
```
2. Customer Billing
3. Exit
Enter choice: 2
Enter customer name: rahul
Enter item number: 109
Enter quantity (Available: 15): 2
Do you want to purchase more? (y/Y/n/N): y
Enter item number: 170
Item not found.
Enter item number: 137
Enter quantity (Available: 30): 33
Error: Insufficient stock.
Enter item number: 155
Enter quantity (Available: 69): 4
Do you want to purchase more? (y/Y/n/N): n

--- BILL ---
Customer: rahul
109 Perfume 2 300.0 600.0
155 cheese 4 56.5 226.0
Grand Total: 826.0
```



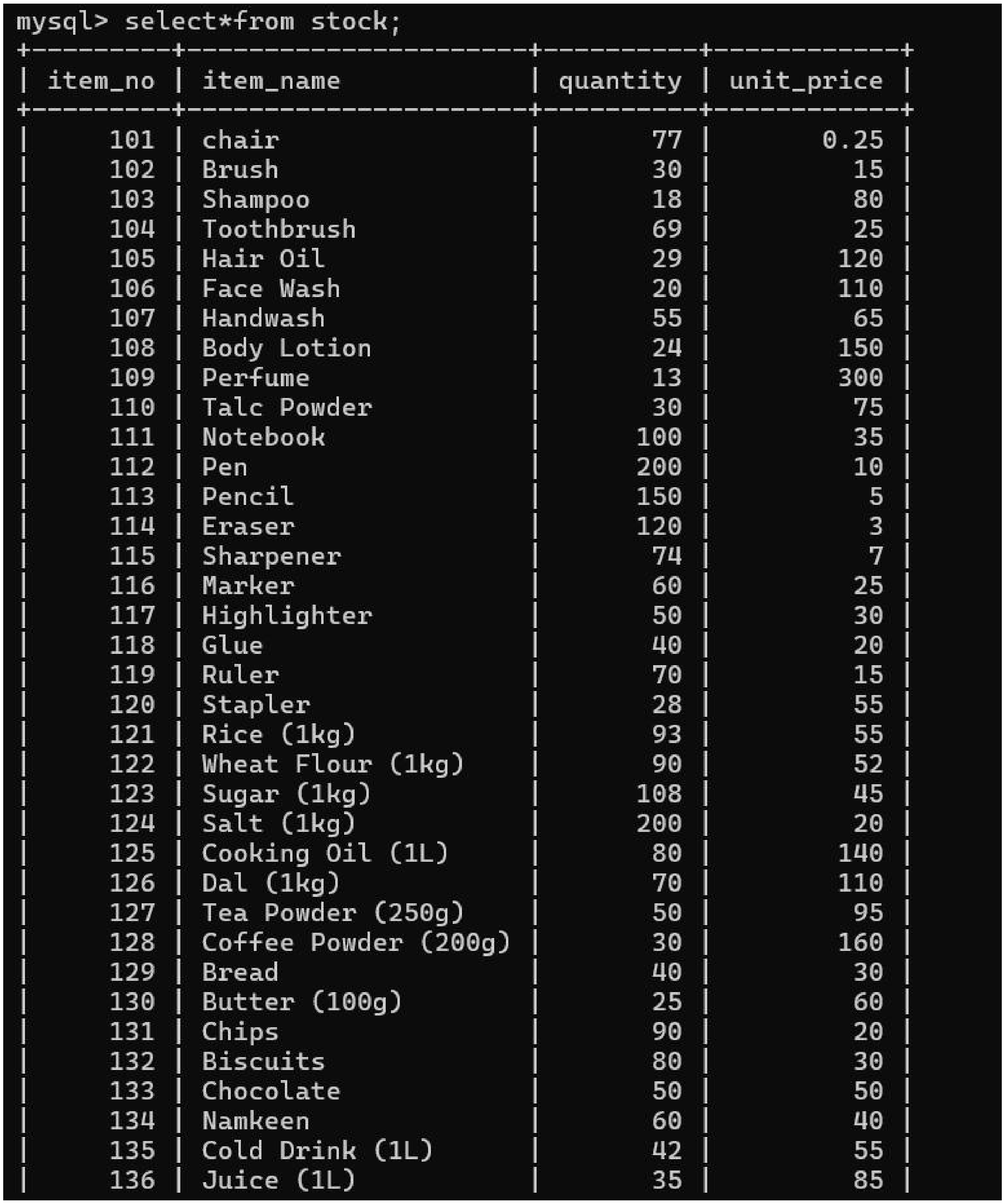
```
mysql> use inventory_db;
Database changed
mysql> show tables;
+-----------------------+
| Tables_in_inventory_db |
+-----------------------+
| bill                  |
| stock                 |
+-----------------------+
2 rows in set (1.93 sec)
```



```
mysql> desc bill;
+---------------+--------------+------+-----+---------+----------------+
| Field         | Type         | Null | Key | Default | Extra          |
+---------------+--------------+------+-----+---------+----------------+
| id            | int          | NO   | PRI | NULL    | auto_increment |
| customer_name | varchar(100) | YES  |     | NULL    |                |
| item_no       | int          | YES  |     | NULL    |                |
| item_name     | varchar(100) | YES  |     | NULL    |                |
| quantity      | int          | YES  |     | NULL    |                |
| unit_price    | float        | YES  |     | NULL    |                |
| total_price   | float        | YES  |     | NULL    |                |
+---------------+--------------+------+-----+---------+----------------+
7 rows in set (1.07 sec)
```



```
mysql> desc stock;
+------------+--------------+------+-----+---------+-------+
| Field      | Type         | Null | Key | Default | Extra |
+------------+--------------+------+-----+---------+-------+
| item_no    | int          | NO   | PRI | NULL    |       |
| item_name  | varchar(100) | YES  |     | NULL    |       |
| quantity   | int          | YES  |     | NULL    |       |
| unit_price | float        | YES  |     | NULL    |       |
+------------+--------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```



```
mysql> select*from stock;
+---------+----------------------+----------+------------+
| item_no | item_name            | quantity | unit_price |
+---------+----------------------+----------+------------+
|     101 | chair                |       77 |       0.25 |
|     102 | Brush                |       30 |         15 |
|     103 | Shampoo              |       18 |         80 |
|     104 | Toothbrush           |       69 |         25 |
|     105 | Hair Oil             |       29 |        120 |
|     106 | Face Wash            |       20 |        110 |
|     107 | Handwash             |       55 |         65 |
|     108 | Body Lotion          |       24 |        150 |
|     109 | Perfume              |       13 |        300 |
|     110 | Talc Powder          |       30 |         75 |
|     111 | Notebook             |      100 |         35 |
|     112 | Pen                  |      200 |         10 |
|     113 | Pencil               |      150 |          5 |
|     114 | Eraser               |      120 |          3 |
|     115 | Sharpener            |       74 |          7 |
|     116 | Marker               |       60 |         25 |
|     117 | Highlighter          |       50 |         30 |
|     118 | Glue                 |       40 |         20 |
|     119 | Ruler                |       70 |         15 |
|     120 | Stapler              |       28 |         55 |
|     121 | Rice (1kg)           |       93 |         55 |
|     122 | Wheat Flour (1kg)    |       90 |         52 |
|     123 | Sugar (1kg)          |      108 |         45 |
|     124 | Salt (1kg)           |      200 |         20 |
|     125 | Cooking Oil (1L)     |       80 |        140 |
|     126 | Dal (1kg)            |       70 |        110 |
|     127 | Tea Powder (250g)    |       50 |         95 |
|     128 | Coffee Powder (200g) |       30 |        160 |
|     129 | Bread                |       40 |         30 |
|     130 | Butter (100g)        |       25 |         60 |
|     131 | Chips                |       90 |         20 |
|     132 | Biscuits             |       80 |         30 |
|     133 | Chocolate            |       50 |         50 |
|     134 | Namkeen              |       60 |         40 |
|     135 | Cold Drink (1L)      |       42 |         55 |
|     136 | Juice (1L)           |       35 |         85 |
```

Testing Approach

Test Cases Included:

1. Add stock with valid/invalid values
2. Delete item not present
3. Billing with insufficient stock
4. Billing with correct quantities
5. Checking bill.csv after multiple purchases

All test cases passed successfully.

Challenges Faced

Maintaining synchronization between stock table and bill updates

Handling invalid input formats

Ensuring bill.csv writes multiple rows correctly

Avoiding crash when incorrect item number entered

Learnings & Key Takeaways

Understanding MySQL–Python integration

Implementing CRUD operations

File handling using CSV module

Error handling and modular programming

Real-world problem-solving through coding