SOURCE CODE:

```
import os
import mysql.connector
from datetime import date
import time

def connect_db():
    return mysql.connector.connect(
        host="localhost",
        username="root",
        password="03april2008",
        database="shopping"
    )

def clear_screen():
    os.system('cls')

def center_print(text):
    print(text.center(60))


#Functions defined for Admin Page

def add_product(cursor, connection):
    center_print("=== Add New Product ===")

    name = input("Enter product name: ")
    category = input("Enter product category: ")
    price = float(input("Enter product price: "))
    product_unit = input("Enter product unit: ")
    stock_qty = int(input("Enter product stock quantity: "))
```

```
    sql = "INSERT INTO products (name, category, price, product_unit,
stock_quantity) VALUES (%s,%s, %s, %s, %s)"
    values = (name, category, price, product_unit, stock_qty)

    cursor.execute(sql, values)
    connection.commit()

    center_print("Product '" + name + "' added successfully.")
    input("Press enter to continue...")

def update_product(cursor, connection):
    center_print("=== Update Product ===")
    product_id = int(input("Enter product ID to update: "))
    cursor.execute("SELECT * FROM products WHERE product_id = %s",
(product_id,))
    product = cursor.fetchone()

    if not product:
        clear_screen()
        center_print("Product not found.")
        return

    center_print("Current details:")
    center_print("Name = " + product[1] + ", Category = " + product[2] +",
Price = " + str(product[3]) + ", Unit = " product[5] + ", Stock = " +
str(product[4]))

    name = input("Enter new name (press enter to skip): ") or product[1]
    category = input("Enter new category (press enter to skip): ") or
product[2]
    price = input("Enter new price (press enter to skip): ") or product[3]
    product_unit = input("Enter new unit (press enter to skip): ") or
product[5]
```

```
    stock_quantity = input("Enter new stock quantity (press enter to skip):
") or product[4]

    sql = "UPDATE products SET name = %s, category = %s, price = %s,
product_unit = %s, stock_quantity = %s WHERE product_id = %s"
    values = (name, category, price, product_unit, stock_quantity,
product_id)

    cursor.execute(sql, values)
    connection.commit()


    center_print("Product ID '" + str(product_id) + "' updated successfully.")
    input("Press enter to continue...")


def delete_product(cursor, connection):
    center_print("=== Delete Product ===")
    product_id = int(input("Enter product ID to delete: "))

    sql = "DELETE FROM products WHERE product_id = %s"

    cursor.execute(sql, (product_id,))
    connection.commit()


    center_print("Product ID '" + str(product_id) + "' deleted successfully.")
    input("Press enter to continue...")


def view_products(cursor):
    center_print("=== Available Products ===")
    print()
```

```
    print("ID".ljust(5) + "Name".ljust(20) + "Category".ljust(15) +
"Price".ljust(10) + "Unit".ljust(7) + "Stock".ljust(10))
    print("-" * 60)

    cursor.execute("SELECT * FROM products")
    products = cursor.fetchall()

    if not products:
        center_print("No products available.")
        return

    for product in products:
        product_id = str(product[0]).ljust(5)
        name = product[1].ljust(20)
        category = product[2].ljust(15)
        price = str(product[3]).ljust(10)
        product_unit=str(product[5]).ljust(7)
        stock = str(product[4]).ljust(10)

        print(product_id + name + category + price + product_unit + stock)

    input("\nPress Enter to continue...")


def view_orders(cursor):
    print()
    center_print("=== Today's Orders ===")
    print()
    today_date = date.today()
    center_print("Date: "+ str(today_date))
    print()
```

```
    cursor.execute("SELECT order_id, customer_name, customer_mobile, total_price, TIME(order_date) as order_time FROM orders WHERE DATE(order_date)= %s",(today_date,))
    orders = cursor.fetchall()

    if not orders:
        center_print("No orders placed today yet.")
    else:
        print("Order ID".ljust(10) + "Customer Name".ljust(15) + "Mobile".ljust(14) + "Total".ljust(12) + "Order Time")
        print("-" * 60)
        total_earnings = 0
        for order in orders:
            total_earnings += order[3]
            print(str(order[0]).ljust(10) + order[1].ljust(15) + order[2].ljust(14) + "Rs. " + str(order[3]).ljust(8) + str(order[4]).ljust(10))

        print("-" * 60)
        center_print("Total Earnings Today: Rs. " + str(round(total_earnings, 2)))

    input("Press Enter to return to the menu...")
```

```
def admin_menu(cursor, connection):
    while True:
        clear_screen()
        center_print("--- Admin Menu ---")
        center_print("1. Add Product")
        center_print("2. Update Product")
        center_print("3. Delete Product")
        center_print("4. View Products")
        center_print("5. View Orders")
        center_print("6. Search Customer Orders")
        center_print("7. Manage Admins")
        center_print("8. Logout")

        choice = input("\nEnter your choice: ")

        if choice == "1":
            add_product(cursor, connection)
        elif choice == "2":
            update_product(cursor, connection)
        elif choice == "3":
            delete_product(cursor, connection)
        elif choice == "4":
            view_products(cursor)
        elif choice == "5":
            view_orders(cursor)
        elif choice == "6":
            search_orders(cursor)
        elif choice == "7":
            manage_admins(cursor, connection)
        elif choice == "8":
            break
        else:
            center_print("Invalid choice. Please try again."
```

```
 input()

# Functions defined for Customer page

cart = []

def view_products_customer(cursor):
    print()
    center_print("=== Available Products ===")
    print()

    print("ID".ljust(5) + "Name".ljust(20) + "Category".ljust(15) +
"Price".ljust(10) + "Unit".ljust(7) )
    print("-" * 60)

    cursor.execute("SELECT * FROM products")
    products = cursor.fetchall()

    # Display each product
    for product in products:
        product_id = str(product[0]).ljust(5)
        name = product[1].ljust(20)
        category = product[2].ljust(15)
        price = str(product[3]).ljust(10)
        product_unit=str(product[5]).ljust(7)

        print(product_id + name + category + price + product_unit)

    input("\nPress Enter to continue...")


def add_to_cart(cursor):
    clear_screen()
```

```
    view_products_customer(cursor)
    while True:

        product_id = int(input("Enter the product ID to add to cart: "))
        quantity = int(input("Enter the quantity: "))

        cursor.execute("SELECT * FROM products WHERE product_id = %s",
(product_id,))
        product = cursor.fetchone()

        if not product:
            center_print("Product not found.")
            input("Press Enter to continue...")
            continue  # Ask for product ID again

        # Check if product is available in stock
        if quantity > product[4]:
            center_print("Not enough stock available.")
            input("Press Enter to continue...")
            continue  # Ask for product ID again

        for item in cart:
            if item['product_id'] == product_id:
                item['quantity'] += quantity
                center_print("Updated quantity of '" + item['name'] + "' to " +
str(item['quantity']) + ".")
                break
        else:
            # Add product to cart
            cart.append({"product_id": product_id, "name": product[1], "price":
product[3], "product_unit":product[5], "quantity": quantity})
            center_print("Added '" + str(quantity) + "' of '" + product[1] + "' to
cart.")
```

```
        ask = input("Do you want to add more items? (yes/no): ").strip().lower()
        if ask != 'yes':
            break

    input("Press Enter to return to the menu...")


def view_cart(cursor):
    print()
    center_print("=== Your Shopping Cart ===")

    if not cart:
        center_print("Your cart is empty.")
        input("Press Enter to continue...")
        return

    # Column headers
    print("ID".ljust(5) + "Name".ljust(20) + "Price".ljust(10) + "Unit".ljust(7) +
"Quantity".ljust(10))
    print("-" * 55)

    for index, item in enumerate(cart):
        product_id = str(index + 1).ljust(5)
        name = item["name"].ljust(20)
        price = str(item["price"]).ljust(10)
        product_unit=str(item["product_unit"]).ljust(7)
        quantity = str(item["quantity"]).ljust(10)

        print(product_id + name + price + product_unit + quantity)

    input("\nPress Enter to continue...")
```

```
def update_cart(cursor):
    print()
    center_print("=== Update Cart ===")

    if not cart:
        center_print("Your cart is empty.")
        input("Press Enter to continue")
        return

    print("Current items in your cart:")

    print("ID".ljust(5) + "Product Name".ljust(20) + "Quantity".ljust(10) +
"Unit".ljust(7) + "Price")
    print("-" * 45)
    for i, item in enumerate(cart, 1):
        print(str(i).ljust(5) + item['name'].ljust(20) +
str(item['quantity']).ljust(10) + str (item['product_unit']).ljust(7) +
str(item['price']))

    item_number = int(input("Enter the item number to update (or 0 to
cancel): ").strip())

    if item_number == 0:
        return

    if 1 <= item_number <= len(cart):
        selected_item = cart[item_number - 1]

        center_print("Selected: " + selected_item['name'] + ", Quantity: " +
str(selected_item['quantity']))
        print("1. Update Quantity")
        print("2. Remove Item")
```

```
        action_choice = input("Enter your choice: ").strip()

        if action_choice == '1':
            new_quantity = int(
                input("Enter new quantity for '" + selected_item['name'] + "' (or 0 to remove): ").strip())
            if new_quantity == 0:
                cart.remove(selected_item)
                center_print("'" + selected_item['name'] + "' removed from the cart.")
            else:
                selected_item['quantity'] = new_quantity
            center_print("Quantity for '" + selected_item['name'] + "' updated to " + str(new_quantity) + ".")

        elif action_choice == '2':
            cart.remove(selected_item)
            center_print("'" + selected_item['name'] + "' removed from the cart.")

        else:
            center_print("Invalid choice. Returning to the menu.")
    else:
        center_print("Invalid item number.")

    input("Press Enter to return to the menu...")



def checkout(cursor,connection):
    print()
    center_print("=== Checkout ===")
```

```
    if not cart:
        center_print("Your cart is empty. Cannot checkout.")
        input("Press Enter to continue...")
        return

    view_cart(cursor)

    customer_name = input("Enter your name: ").strip()
    customer_mobile = input("Enter your mobile number: ").strip()

    if customer_name =="" or customer_mobile == "":
        center_print("Enter proper details and content")
        input("Please enter to continue .. ")
        return


    total_price = sum(item["price"] * item["quantity"] for item in cart)
    gst= float(total_price) * float(0.18)
    discount= float(total_price) * float(0.10)

    final_total= float(total_price) + float(gst) - float(discount)
    cursor.execute("INSERT INTO orders (customer_name,
customer_mobile, total_price)VALUES (%s, %s, %s)", (customer_name,
customer_mobile, final_total))

    connection.commit()

    order_id = cursor.lastrowid

    for item in cart:
```

```
        cursor.execute("INSERT INTO order_items (order_id, product_name,
quantity, price)VALUES (%s, %s, %s, %s)", (order_id, item['name'],
item['quantity'], item['price']))

    connection.commit()

    center_print("Your Total: " + str(final_total) + " Rs")
    confirm = input("Confirm checkout? (yes/no): ").strip().lower()

    if confirm == "yes":
        center_print("Checkout successful. Thank you for your purchase!")
        print_bill(customer_name, customer_mobile, total_price, gst,
discount, final_total)
        cart.clear()
    else:
        center_print("Checkout cancelled.")

    input("Press Enter to return to the menu...")


def print_bill(customer_name, customer_mobile, total_price, gst,
discount, final_total):
    clear_screen()
    center_print("=" * 60)

    center_print("SHOPPING MANAGEMENT SYSTEM")
    center_print("=" * 60)

    center_print(("Customer: " + customer_name))
    center_print(("Mobile: " + customer_mobile))
    center_print(("Date: " + time.strftime('%Y-%m-%d %H:%M:%S')))
    center_print("-" * 60)
```

```
    print("Product Name".ljust(20) + "Quantity".ljust(10) + "Price".ljust(10) + "Total".ljust(10))
    center_print("-" * 60)

    for item in cart:
        total_item_price = item['quantity'] * item['price']
        print(item['name'].ljust(20) + str(item['quantity']).ljust(3) + str(item["product_unit"]).ljust(7) + "Rs. " + str(item['price']).ljust(10) + "Rs. " + str(total_item_price).ljust(8))

    print("-" * 60)

    print("Subtotal:".ljust(40) + "Rs. " + str(round(total_price, 2)).ljust(10))
    print("GST (18%):".ljust(40) + "Rs. " + str(round(gst, 2)).ljust(10))
    print("Discount:".ljust(40) + "Rs. " + str(round(discount, 2)).ljust(10))

    print("=" * 60)
    print("Total Amount Payable: Rs. " + str(round(final_total, 2)))
    print("=" * 60)

    center_print("\nThank you for shopping with us!")

def search_orders(cursor):
    clear_screen()
    center_print("=== Search Customer Orders ===")
    print()
    mobile_number = input("Enter mobile number: ").strip()

    cursor.execute("SELECT order_id, customer_name, total_price, DATE(order_date) FROM orders WHERE customer_mobile = %s", (mobile_number,))

    orders = cursor.fetchall()
```

```
    if not orders:
        center_print("No orders found for this mobile number.")
        input("Press Enter to return to the menu...")
        return
    print()
    center_print("=== Order History ===")
    print("Order ID".ljust(15) + "Customer Name".ljust(20) + "Total
Price".ljust(15) + "Order Date")
    print("-" * 60)

    for order in orders:
        print(str(order[0]).ljust(15) + order[1].ljust(20) + ("Rs. " +
str(order[2])).ljust(15) + str(order[3]))
    print("-" * 60)
    print()
    order_id = input("Enter Order ID to view details (or press Enter to go
back): ").strip()

    if order_id:
        view_order_details(cursor, order_id)
    else:
        input()


def view_order_details(cursor, order_id):
    print()
    center_print("=== Order Details ===")
    print()
    cursor.execute("SELECT order_items.product_name,
order_items.quantity, order_items.price FROM order_items WHERE
order_items.order_id = %s", (order_id,))
    items = cursor.fetchall()
```

```
    print("Product Name".ljust(20) + "Quantity".ljust(10) + "Price".ljust(10) +
"Total".ljust(10))
    print("-" * 60)

    for item in items:
        total_item_price = item[1] * item[2]
        print(item[0].ljust(20) + str(item[1]).ljust(10) +
              "Rs. " + str(item[2]).ljust(8) + "Rs. " + str(total_item_price).ljust(8))

    print("-" * 60)
    print()
    input("Press Enter to return to the menu...")


def customer_menu(cursor,connection):
    while True:
        clear_screen()
        center_print("--- Customer Menu ---")
        center_print("1. View Products")
        center_print("2. Add to Cart")
        center_print("3. View Cart")
        center_print("4. Update Cart")
        center_print("5. Checkout")
        center_print("6. Search Previous Orders")
        center_print("7. Exit")

        choice = input("\nEnter your choice: ")

        if choice == "1":
            view_products_customer(cursor)
        elif choice == "2":
            add_to_cart(cursor)
```

```
        elif choice == "3":
            view_cart(cursor)
        elif choice == "4":
            update_cart(cursor)
        elif choice == "5":
            checkout(cursor,connection)
        elif choice=="6":
            search_orders(cursor)
        elif choice =="7":
            break
        else:
            center_print("Invalid choice. Please try again.")
            input()


#Function for welcome page and login

def admin_login(cursor):

    clear_screen()
    center_print("=== Admin Login ===")

    username = input("Enter admin username: ")
    password = input("Enter admin password: ")

    cursor.execute("SELECT * FROM admin_users WHERE username = %s
AND password = %s", (username, password))
    admin = cursor.fetchone()

    if admin:
        center_print("Login successful!")
        input("Press Enter to continue...")
        return True
```

```
    else:
        center_print("Invalid username or password.")
        input("Press Enter to continue...")
        return False

def manage_admins(cursor, db):
    while True:
        clear_screen()
        center_print("=== Manage Admins ===")
        center_print("1. Add Admin")
        center_print("2. Change Password")
        center_print("3. Delete Admin")
        center_print("4. Return to Admin Menu")

        choice = input("Enter your choice: ").strip()

        if choice == '1':
            add_admin(cursor, db)
        elif choice == '2':
            change_admin_password(cursor, db)
        elif choice == '3':
            delete_admin(cursor, db)
        elif choice == '4':
            break
        else:
            center_print("Invalid choice. Please try again.")
            input()

def delete_admin(cursor, db):
    print()
    username = input("Enter the admin username to delete: ")
```

```
    cursor.execute("DELETE FROM admin_users WHERE username = %s",
(username,))
    db.commit()
    center_print("Admin user '" + username + "' deleted successfully.")
    input("Press enter to continue...")

def change_admin_password(cursor, db):
    print()
    username = input("Enter admin username: ")
    old_password = input("Enter current password: ")

    # Check if the admin exists
    cursor.execute("SELECT * FROM admin_users WHERE username = %s
AND password = %s", (username, old_password))
    admin = cursor.fetchone()

    if admin:
        new_password = input("Enter new password: ")

        # Update the password in the database
        cursor.execute("UPDATE admin_users SET password = %s WHERE
username = %s", (new_password, username))
        db.commit()
        center_print("Password changed successfully.")
        input("Press enter to continue...")
    else:
        center_print("Invalid username or current password.")
        input("Press enter to continue...")

def add_admin(cursor, db):
    print()
    username = input("Enter new admin username: ")
    password = input("Enter new admin password: ")
```

```
    cursor.execute("INSERT INTO admin_users (username, password)
VALUES (%s, %s)", (username, password))
    db.commit()
    center_print("Admin user '" + username + "' added successfully.")
    input("Press enter to continue...")

def welcome_page():
    clear_screen()
    center_print("=" * 55)

    center_print(" WELCOME TO THE SHOPPING MANAGEMENT SYSTEM ")
    center_print("-" * 55)  # Separator line
    center_print(" Created by: Study Stackers")
    center_print(" Class: COMPUTER SCIENCE XII PCM ")
    center_print("=" * 55)

    time.sleep(1)
    center_print("\n")
    center_print("Please select an option:")
    center_print("1. Admin Login")
    center_print("2. Customer Page")
    center_print("3. Exit")

    choice = input("\nEnter your choice: ")
    return choice

def main():
    # Connect to MySQL
    connection = connect_db()
    cursor = connection.cursor()

    while True:
```

```
    choice = welcome_page()

    if choice == "1":
        if admin_login(cursor):
            admin_menu(cursor, connection)
    elif choice == "2":
        customer_menu(cursor,connection)
    elif choice == "3":
        center_print("Thank you for using the system!")
        time.sleep(5)
        break
    else:
        center_print("Invalid choice. Please try again.")
        input("Press Enter to continue...")

  cursor.close()
  connection.close()

if __name__ == "__main__":
  main()
```