Code

main.py

```python
from boxprint import box
import error_corrector as ec

print("""
888b    888          888    888      d8b
8888b   888          888    888      Y8P
88888b  888          888    888
888Y88b 888  .d88b.  888888 88888b.  888 88888b.   .d88b.
888 Y88b888 d88""88b 888    888 "88b 888 888 "88b d88P"88b
888  Y88888 888  888 888    888  888 888 888  888 888  888
888   Y8888 Y88..88P Y88b.  888  888 888 888  888 Y88b 888
888    Y888  "Y88P"   "Y888 888  888 888 888  888  "Y88888
                                                       888
We sell everything...                             Y8b d88P
                                                   "Y88P" """)

print("Starting...")
ec.run_checks()

box(["Please Log In"], width=20)
user = input("Select User [Customer/Admin]: ")

if user.lower() == "customer" or user.lower() == "c":
    import customer
elif user.lower() == "admin" or user.lower() == "a":
    import admin
else:
    print("User not defined")
```

error_corrector.py

```python
def run_checks():
    """Check whether MySQL is installed, it is accessible
    and whether it has the proper databases and tables"""

    # Check for MySQL-Python connector
    try:
        import mysql.connector
        print("[ ok ] mysql.connector working")
    except:
        print("[Error] Unable to import mysql.connector")
        exit()

    # Check for MySQL installation
    try:
        conn = mysql.connector.connect(host="localhost", user="root",
passwd="password")
        print("[ ok ] MySQL found")
    except:
        print("[Error] Unable to connect to MySQL")
        exit()

    # Check for database
    try:
        cursor = conn.cursor()
        cursor.execute("use nothing_shop")
        print("[ ok ] Database found")
    except:
        print("[Error] Unable to access database")
        exit()

    # Check for tables
    try:
        cursor.execute("select * from products")
        print("[ ok ] Tables found")
    except:
        print("[Error] Unable to access table")
        exit()

    # Check for CSV module
    try:
        import csv
        print("[ ok ] CSV module found")
    except:
        print("[Error] Unable to import csv")
        exit()
```

sql_handler.py

```python
import mysql.connector
from boxprint import box

conn =
mysql.connector.connect(host="localhost",user="root",passwd="password",database="nothi
ng_shop")
cur = conn.cursor()

# Some General Functions
def getPIDs():
    cur.execute("select pid from products")
    data = cur.fetchall()
    pids = []

    for item in data:
        pids.append(item[0])

    return pids

def getShop():
    cur.execute("select * from products")
    products = cur.fetchall()
    return products

def showShop():
    products = getShop()
    print(products)

    if products == []:
        box(["Shop is Empty"], width=5)

    else:
        for item in products:
            box([
            f"ID: {item[0]}",
            f"Name  :  {item[1]}",
            f"Price :  {item[2]}",
            f"Stock :  {item[3]}",
            ], width=30)

def getProduct(pid):
    cur.execute(f"select * from products where pid={pid}")
    product = cur.fetchall()

    if product:
        return product[0]
    else:
        return False

# Admin Interface
```

```python
def listProduct(name, price, stock):
    pids = getPIDs()

    for n in range(1, len(pids)+2):
        if n not in pids:
            newpid = n
            break

    try:
        cur.execute(f"insert into products values({newpid}, '{name}', {price},
{stock})")
        conn.commit()

    except:
        print("[ INVALID PARAMETERS ]")
        return False

    return getProduct(newpid)

def unlistProduct(pid):
    pids = getPIDs()

    if pid in pids:
        cur.execute(f"delete from products where pid={pid}")
        conn.commit()
        return True

    else:
        print("[ INVALID PRODUCT ID ]")
        return False

def modifyProduct(pid, name, price, stock):

    if pid in getPIDs():
        cur.execute(f"update products set name='{name}', price={price}, stock={stock}
where pid={pid}")
        conn.commit()

        return getProduct(pid)

    else:
        print("[ INVALID PRODUCT ID ]")
        return False
```

boxprint.py

```python
charset1 = {"tr":"┓", "tl":"┏", "br":"┛", "bl":"┗", "vr":"┃", "hr":"━"}
charset2 = {"tr":"╗", "tl":"╔", "br":"╝", "bl":"╚", "vr":"║", "hr":"═"}
charset3 = {"tr":"╮", "tl":"╭", "br":"╯", "bl":"╰", "vr":"│", "hr":"─"}

chars = charset3

def box(lines, width=40):
    """Print the output neatly in a box"""

    # if line is longer that width, then set it as width
    for line in lines:
        if len(line)>width:
            width = len(line)

    # pad the right of all lines with spaces
    newlines = []
    for line in lines:
        newlines.append(line+ " "*(width-len(line)))

    # print lines in a box
    print(chars["tl"] + chars["hr"]*(width+2) +chars["tr"]) # print top of box
    for line in newlines:
        print(chars["vr"]+" " + line + " "+chars["vr"])
    print(chars["bl"] + chars["hr"]*(width+2) +chars["br"]) # print bottom of box

def pad(string, length):
    string = str(string)
    if len(string) > length:
        return string
    else:
        new_string = string + (length-len(string))*" "
        return new_string
```

admin.py

```python
from boxprint import box
import sql_handler as sqh

box(["Welcome Admin"], width=20)

while True:
    print("[ADMIN] q:QUIT l:LIST-ITEM u:UNLIST-ITEM m:MODIFY-ITEM s:SHOW-SHOP")
    ch = input(": ")

    if ch=="l":
        name = input("Enter product name: ")
        price = float(input("Enter product price: "))
        stock = int(input("Enter product stock: "))

        product = sqh.listProduct(name, price, stock)
        if product:
            box([f"{product[1]} added to shop"])

    if ch=="u":
        pid = int(input("Enter product id: "))

        response = sqh.unlistProduct(pid)

        if response:
            box([f"Product {pid} was removed from shop"])

    if ch=="m":
        pid = int(input("Enter product id: "))

        print("Enter the new details")
        name = input("  Name :")
        price = float(input("  Price:"))
        stock = int(input("  Stock:"))

        product = sqh.modifyProduct(pid, name, price, stock)

        if product:
            box([f"{product[1]} was modified"], width=5)

    if ch=="s":
        sqh.showShop()

    if ch=="q":
        print("[ Exiting ]")
        exit()
```

customer.py

```python
from boxprint import box
import sql_handler as sqh
import cart

box(["Welcome Customer"], width=20)

box(["Shop"], width=10)

while True:
    print("[SHOP] q:QUIT a:ADD c:GO-TO-CART s:SHOW-SHOP")
    ch = input(": ")

    if ch=="a":
        cart.addProduct()

    if ch=="c":
        cart.cart_init()

    if ch=="s":
        sqh.showShop()

    if ch=="q":
        print("[ Exiting ]")
        exit()
```

cart.py

```python
from boxprint import box, pad
import sql_handler as sqh
import csv

user_cart = []

def cart_init():

    while True:

        box(["Cart"], width=10)

        showCart()

        print("[CART] q:QUIT r:REMOVE x:EXPORT")
        ch = input(": ")

        if ch=="r":
            removeProduct()

        elif ch=="x":
            exportAsCSV()

        elif ch=="q":
            print("[ Returning to SHOP ]")
            break

        else:
            print("[ INVALID INPUT ]")

def addProduct():
    pid = int(input("Enter the product id: "))
    pids = sqh.getPIDs()

    if pid in user_cart:
        box(["Product already in cart"], width=5)

    elif pid in pids:
        user_cart.append(pid)
        product = sqh.getProduct(pid)
        box([f"{product[1]} added to cart"], width=5)

    else:
        print("[ INVALID PRODUCT ID ]")

def removeProduct():
    pid = int(input("Enter the product id: "))

    if pid in user_cart:
        user_cart.remove(pid)
        product = sqh.getProduct(pid)
        box([f"{product[1]} removed from cart"], width=5)

    else:
        print("[ PRODUCT NOT IN CART ]")

def showCart():
    if user_cart == []:
        print("[ CART IS EMPTY ]")

    else:

        lines = []

        lines.append("ID" + "  "+ "Name" + " "*28 + "Price"+ " "*5)
        lines.append("-"*46)

        price_total = 0

        for pid in user_cart:
            product = sqh.getProduct(pid)
            lines.append(f"{pad(product[0], 2)}  {pad(product[1], 30)}
{pad(product[2], 10)}") # len -> 46
            price_total+=product[2]

        box(lines)
        box([f"Total: {price_total}"], width=28)

def exportAsCSV():
    name = input("Enter name of file (without .csv): ")
    with open(name+".csv", "w", newline="") as f:
        wr = csv.writer(f)
        wr.writerow(["Product ID", "Name", "Price"])

        for pid in user_cart:
            product = sqh.getProduct(pid)
            wr.writerow([product[0], product[1], product[2]])
    box([f"Cart was exported to {name}.csv"], width=5)
```