```
from datetime import datetime
from pandas import pandas as pd
import pdfkit as pdf
import support as sup
import num2words as n2w
import os

# to add path variable
config = pdf.configuration(wkhtmltopdf="C:/Program
Files/wkhtmltopdf/bin/wkhtmltopdf.exe")

name = []
Mrp = []
final = []
Gst = []
discamt = []
gstamt = []
cod = []
qty = []
disc = []
xcode = []
ch = []
ch1 = []
xname = []
xqty = []
bill_no = 1
chkcode = []
status = []
chkqty = []


# function to add new product
def addnewprod():
    _name = str(input("Enter product name:  "))
    _code = str(input("Enter product code: "))
    _MRP = str(input("Enter MRP of the product:"))
    _GST = str(input("Enter GST of the product(in %):"))
    _qty = str(input("Enter quantity available"))

    file = open("G:/billing/itemlist.txt", "a")  # open a new file
    file.writelines([_code, "*", _name, "*", _MRP, "*", _GST, "*", _qty, "\n"])  #
to add details int file
    file.close()
    print("Data has been saved SUCCESSFULLY")


# function to display products available
def readproducts():
```

```
    file_open = open("G:/billing/itemlist.txt", "r")
    fh = file_open.readlines()  # To read the lines in the file
    print("-" * 56)
    for index in range(0, len(fh)):
        x1, x2, x3, x4, x5 = fh[index].split("*")
        xcode.append(x1)  # Assigning the values to the required list
        chkcode.append(x1)
        xname.append(x2)
        xqty.append(int(x5))
        if int(x5) != 0:
            status.append("In stock")
        else:
            chkqty.append(x1)
            status.append("No stock")
    data = {"product code": xcode, "product name": xname, "Quantity available":
xqty, "Status": status}
    for i in range(0, len(xcode)):
        ch.append(i + 1)
    fh = pd.DataFrame(data, index=ch)  # To display the available options
    print(fh)
    print("-" * 56)

    # To clear the list
    xcode.clear()
    xname.clear()
    xqty.clear()
    ch.clear()
    status.clear()
    file_open.close()


# function to accept products and quantities
def quantites():
    while True:
        x = (str(input("Enter product code or press q to obtain bill ")))
        if len(x) == 0:
            print("please find something to add ")
            continue

        if (len(cod) == 0) and (x == "q"):
            print("Cart is empty")
            continue

        if x == "q":
            chkqty.clear()
            chkcode.clear()
            break
```

```
        if x not in chkcode:
            print('Enter a valid code')
            continue

        if x in chkqty:
            print('Stock not available')
            continue

        else:
            cod.append(x)
        try:
            qty.append(int(input("Enter quantity ")))
        except ValueError:
            qty.append(int(input("Enter proper value ")))

        try:
            disc.append(int(input("Enter percentage of discount applicable ")))
        except ValueError:
            disc.append(int(input("Enter proper value ")))


# Function to calculate cost,discount,total cost
def calculate():
    file_app = open("G:/billing/itemlist.txt", "r")
    fh = file_app.readlines()
    file_app.close()
    print('-' * 29, end='')
    print('Charles Super Market', end='')
    print('-' * 30)
    print('Billing Date/Time:', datetime.now().strftime('%d-%m-%Y/%I:%M:%S %p'),
end='')
    print(' ' * 26 + 'Bill No:', bill_no)
    print('-' * 79)

    # computation of the data obtained
    for i in range(0, len(cod)):
        for index in range(0, len(fh)):
            x1, x2, x3, x4, x5 = fh[index].split("*")
            if cod[i] == x1:
                name.append(x2)
                Mrp.append(x3)
                y4 = int(x4)
                y5 = int(x5)
                Gst.append(y4)
                price = (qty[i] * float(x3))  # price calculations
                discount_price = price * (disc[i] / 100)  # discount calculations
                discamt.append(discount_price)
                totprice = price - discount_price
```

```
                gst_price = price * (int(x4) / 100)  # Gst calculations
                gstamt.append(gst_price)
                grand_price = totprice + gst_price
                final.append(float(grand_price))
                qtyleft = y5 - qty[i]
                sup.replaceline(x1, x2, x3, x4, qtyleft)
                break

    # To print processed data in a formatted way(table)
    printdata = {"Product name": name, "Product code": cod, "Quantity": qty, "MRP":
Mrp, "GST(in %)": Gst,
                 "Discount(in %)": disc, "Price": final}
    for i in range(0, len(cod)):
        ch1.append(i + 1)
    pd.reset_option('display.max_columns', None)
    pd.reset_option('display.max_rows', None)
    table = pd.DataFrame(printdata, index=ch1)
    print(table)

    # Concluding the bill
    print('-' * 79)
    print("Grand total:             ", sup.currency(round(sum(final), 2)))
    print("Total saved amount is:   ", sup.currency(round(sum(discamt), 2)))
    print("Total GST amount is:     ", sup.currency(round(sum(gstamt), 2)))
    print("*" * 79)
    print('In Words: ', n2w.num2words(round(sum(final)), lang="en"), ' Rupees Only',
sep='')
    print('*' * 79)
    print("NOTE : Goods once sold will not be Replaced/Refunded")
    print("Thankyou for shopping with us")
    print('*' * 79)

    # To generate a pdf file
    table.to_html(sup.file_name_html(cust_name), )
    file = open(sup.file_name_html(cust_name), "a")
    file.writelines("<br>")
    file.writelines("*" * 79)
    file.writelines("<br>")
    file.writelines(["Grand total:             ",
str(sup.currency1((round(sum(final), 2)))), "  Rupees", "<br>"])
    file.writelines("<br>")
    file.writelines(["Total saved amount is:   ",
str(sup.currency1(round(sum(discamt), 2))), "  Rupees", "<br>"])
    file.writelines("<br>")
    file.writelines(["Total GST amount is:     ",
str(sup.currency1(round(sum(gstamt), 2))), "  Rupees", "<br>"])
    file.writelines("*" * 79)
    file.writelines("<br>")  # adding extra lines
```

```
    file.writelines("<br>")
    file.writelines("<br>")
    file.writelines("<br>")
    file.writelines("<br>")
    file.writelines("<br>")
    file.writelines([" "*50,"Signature"])
    file.close()
    # Customize the PDF
    options = {'quiet': '', 'page-size': 'Letter', 'margin-top': '1.5in',
'margin-left': '1.5in',
               'margin-bottom': '1.5in', 'margin-right': '1in'
        , 'encoding': "UTF-8", 'custom-header': [('Accept-encoding', 'gzip')],
               '--header-html': 'G:/billing/htmlfiles/header.html'
        , 'no-outline': False}
    pdf.from_file(sup.file_name_html(cust_name), sup.file_name_pdf(cust_name),
configuration=config, options=options)
    print("Generated bill is saved SUCESSFULLY in main server")
    os.remove(sup.file_name_html(cust_name))

    # to clear the unwanted data
    name.clear()
    Mrp.clear()
    final.clear()
    Gst.clear()
    discamt.clear()
    gstamt.clear()
    cod.clear()
    qty.clear()
    disc.clear()
    ch1.clear()

# Code begins from here
while True:
    try:
        c = int(input("Press 1 to add new product\n" "Press 2 for new bill\n"
                  "Press 3 to update stock for existing product\n" "Press 4 to
EXIT\n"))
    except ValueError:
        c = int(input("Enter proper value"))

    if c == 1:
        addnewprod()
    elif c == 2:
        readproducts()
        cust_name = str(input("Enter customer name: "))
        quantites()
        calculate()
        bill_no += 1
```

```
elif c == 3:
    readproducts()
    COD = str(input("Product code: "))
    try:
        QTY = int(input("Quantity: "))
    except ValueError:
        QTY = int(input("Enter proper value "))
    sup.Replaceqty(COD , QTY)
elif c == 4:
    print("Have a nice day!!!!")
    break
else:
    print("Invalid option Restart your application")
    exit(1)
```